Deep Symbolic Regression for Physics Guided by Units Constraints: Toward the Automated Discovery of Physical Laws

Wassim Tenachi, Rodrigo Ibata, Foivos I. Diakogiannis

The Astrophysical Journal, 959:99 (20pp), 2023 December 20 https://doi.org/10.3847/1538-4357/ad014c

Deep Symbolic Regression for Physics Guided by Units Constraints: Toward the Automated Discovery of Physical Laws

Wassim Tenachi[1], Rodrigo Ibata[1], and Foivos I. Diakogiannis[2]
[1] Université de Strasbourg, CNRS, Observatoire astronomique de Strasbourg, UMR 7550, F-67000 Strasbourg, France
[2] Data61, CSIRO, Kensington, WA 6155, Australia
Received 2023 March 3; revised 2023 October 6; accepted 2023 October 6; published 2023 December 11

Abstract

Symbolic regression (SR) is the study of algorithms that automate the search for analytic expressions that fit data. While recent advances in deep learning have generated renewed interest in such approaches, the development of SR methods has not been focused on physics, where we have important additional constraints due to the units associated with our data. Here we present Φ-SO, a physical symbolic optimization framework for recovering analytical symbolic expressions from physics data using deep reinforcement learning techniques by learning units constraints. Our system is built, from the ground up, to propose solutions where the physical units are consistent by construction. This is useful not only in eliminating physically impossible solutions but also because the *grammatical* rules of dimensional analysis enormously restrict the freedom of the equation generator, thus vastly improving performance. The algorithm can be used to fit noiseless data, which can be useful, for instance, when attempting to derive an analytical property of a physical model, and it can also be used to obtain analytical approximations of noisy data. We test our machinery on a standard benchmark of equations from the Feynman Lectures on Physics and other physics textbooks, achieving state-of-the-art performance in the presence of noise (exceeding 0.1%) and show that it is robust even in the presence of substantial (10%) noise. We showcase its abilities on a panel of examples from astrophysics.

Unified Astronomy Thesaurus concepts: Neural networks (1933); Regression (1914); Astronomy data modeling (1859); Algorithms (1883)

1. Introduction

Galileo famously intuited in Opere Il Saggiatore (Galilei 1623) that the book of the Universe "è scritto in lingua matematica" (i.e., it is written in mathematical language). Ever since, it has been a central concern of physics to attempt to explain the properties of nature in mathematical terms, by proposing or deriving mathematical expressions that encapsulate our measurements from experiment and observation. This approach has proven to be immensely powerful. Through trial and error over the centuries, the great masters of physics have developed and bequeathed us a rich toolbox of techniques that have allowed us to understand the world and build our modern technological civilization. But now, thanks to the development of modern deep learning networks, there is hope that this endeavor could be accelerated by making use of the fact that machines are able to survey a vastly larger space of trial solutions than an unaided human.

Of course, since the beginning of the computer revolution, many methods have been developed to fit coefficients of linear or nonlinear functions to data (see, e.g., Press et al. 2007). While such approaches are undoubtedly very useful, the procedures we wish to discuss in the present contribution are more general, in the sense that they aim to find the functions themselves, as well as any necessary fitting coefficients. In particular, we wish to infer a free-form symbolic analytical function $f: \mathbb{R}^n \longrightarrow \mathbb{R}$ that fits $y = f(\boldsymbol{x})$ given $(\boldsymbol{x}, y)$ data. In computer science, these procedures are generally referred to as symbolic regression (SR).

1.1. Motivations from Physics and Big Data

Although there are multiple demonstrations of the capabilities of SR in physics (e.g., Wu & Tegmark 2019; Liu & Tegmark 2021; Liu et al. 2021; Reinbold et al. 2021; DiPietro & Zhu 2022; Lemos et al. 2022; Bartlett et al. 2023) and astrophysics (e.g., Wadekar et al. 2020; Delgado et al. 2022; Matchev et al. 2022; Shao et al. 2022; Wong & Cranmer 2022; Desmond et al. 2023; Wadekar et al. 2023), to date, SR has never been used to discover new physical laws from astrophysical measurements. Yet this may change thanks to new observational missions and surveys such as Gaia (Gaia Collaboration et al. 2016), Euclid (Laureijs et al. 2011), LSST (Collaboration 2009; Željko et al. 2019), and SKA (Carilli & Rawlings 2004). With these and other large surveys, our field is entering a new era of data abundance, and there is considerable excitement at the possibility of identifying new empirical laws from these unprecedentedly rich and intricate data sets that could eventually lead to the discovery of new physics. However, the colossal amount of data also presents significant conceptual challenges. Although deep learning will allow us to extract valuable information from large surveys, it is both blessed and plagued by the underlying neural networks that are one of its most potent components. Neural networks are flexible and powerful enough to model any physical system (that can be described as a Lebesgue integrable function, Lu et al. 2017) and work in high dimensions, but they unfortunately largely consist of non-interpretable black boxes. Clearly, interpretability and intelligibility are of great importance in physics, which begs the question, How can one

harness information from these large data sets while retaining their ability to interpret and connect with theory? After training a deep neural network to fit a data set, can one open the black box, to understand the physics modeled inside?

1.2. SR

SR addresses these issues by producing compact, interpretable, and generalizable models. Indeed, the goal is to find very simple prescriptions such as Newton's law of universal gravitation that can well explain a vast number of experiments and observations. There are many advantages to discovering physical laws in the form of succinct mathematical expressions rather than large numerical models:

1. Compactness: SR methods can produce extremely compact models, e.g., with expressions containing $\sim 10^1$ symbols (La Cava et al. 2021), which is on a par with the typical length of expressions in the Feynman Lectures on Physics (Feynman et al. 1971), for example, which is of 16 (with the higher end of SR methods producing expressions well below a length of 10^3). In contrast, numerical models, such as neural networks, typically rely on many more parameters. This makes the models computationally inexpensive to run and in principle also enables SR to correctly recover the exact underlying mathematical expression of a data set using much less data than traditional machine-learning approaches (Wilstrup & Kasak 2021) and with a robustness toward noise even for perfect model recovery (La Cava et al. 2021; Reinbold et al. 2021).
2. Generalization: In addition, unless the target equations consist of arbitrarily long polynomials, the compact expressions produced by SR are less prone to overfitting on measurement errors and are much more robust and reliable outside of the fitting range provided by the data than large numerical models, showing overall much better generalization capabilities as demonstrated in Sahoo et al. (2018), Wilstrup & Kasak (2021), Kamienny & Lamprier (2022), and Kamienny et al. (2022; we will provide an example of this in Section 5.5). This makes SR a potentially powerful tool for discovering the most concise and general representation of the measurements.
3. Intelligibility and interpretability: Since the models produced by SR consist of mathematical expressions, their behavior is intelligible to us, unlike large numerical models. This is of enormous value in physics (Wu & Tegmark 2019) as SR models may enable one to connect newly discovered physical laws with theory and make subsequent theoretical developments. More broadly, this approach fits into the increasing push toward intelligible (Sabbatini & Calegari 2022), explainable (Arrieta et al. 2020), and interpretable (Murdoch et al. 2019) machine-learning models, which is especially important in fields where such models can affect human lives.[3,4]

However, although the prospect of using SR for discovering new physical laws may be very appealing, it is also extremely challenging to implement. It is useful to consider the difficulty of this problem if one were to approach it in a naive way. Suppose in the trial analytic expressions, we allow for an expression length of 35 symbols (as we will do below), and that there are 15 different variables or operations (e.g., x, $+$, $-$, $\times$, $/$, sin, log, ...) to choose from for each symbol (which is on a par with what we will do below). A naive brute force attempt to fit the data set might then have to consider up to $15^{35} \approx 1.5 \times 10^{41}$ trial solutions, which are obviously vastly beyond our computational means to test against the data at the present day or at any time in the foreseeable future, making SR an *NP hard* (nondeterministic polynomial time) problem (Virgolin & Pissis 2022). Furthermore, one has to account for the optimization of free constants in the proposed expressions. The obvious conclusion one draws from these considerations is that SR requires one to develop highly efficient strategies to prune poor guesses.

1.3. Physical SR

There are multiple approaches to SR (detailed in Section 2) that are capable of generating accurate analytical models. However, in the context of physics, we have the additional requirement that our equations must be balanced in terms of their physical units, as otherwise, the equation is simply nonsensical, irrespective of whether it gives a good fit to the numerical values of the data. Although powerful, to the best of our knowledge, all of the available SR approaches spend most of their time exploring a search space where the immense majority of candidate expressions are unphysical in terms of units and thus often end up producing unphysical models (with the exception of approaches in which variables are rendered dimensionless beforehand as discussed in Section 3.2). A very simple solution to this problem would have been to use an existing SR code, and check post hoc whether the proposed solutions obey that constraint. Not only does that constitute an immense waste of time and computing resources, which could render many interesting SR tasks impossible, it also makes a significant fraction of the resulting *best* analytical models unusable and uninterpretable. We note that for the sake of clarity, throughout this paper we refer to a system of unique quantities such as physical dimensions $\{L, M, T, I, \Theta, N, J\}$, i.e., with physical units {m, kg, s, A, K, mol, cd} a subset thereof, or problem-specific quantities such as $\{L, V, \rho, P, v\}$, i.e., with physical units $\{\mathrm{m}, \mathrm{m}^3, \mathrm{kg\,m^{-3}}, \mathrm{Pa}, \mathrm{m \cdot s^{-1}}\}$ as *units*[5].

At first glance, one could think of units constraints as severe restrictions that limit the capabilities of SR as they would prevent the generation of unphysical intermediary expressions. However, in this work, we show that respecting physical constraints actually helps improve SR performance not only in terms of interpretability but also in accuracy by guiding the exploration of the space of solutions toward exact analytical laws. This is consistent with the studies of Kammerer et al. (2020) and Petersen et al. (2021a, 2021b), who found that using in situ constraints during analytical expression generation is much more efficient as it vastly reduces the search space of trial expressions (though we note that incorporating such constraints in those frameworks would not be straightforward as one would need to recompute the whole relational graph representing an analytical expression and its underlying units constraints each time a new symbol is added).

Here we present our physical symbolic optimization framework (Φ-SO), which was designed from the beginning to

[3] https://www.congress.gov/bill/117th-congress/house-bill/6580/
[4] https://artificialintelligenceact.eu/

[5] Although this can also be extended to systems with nonphysical quantities, such as {scalar, vector, matrix} or even {dollars, capita, annum}.

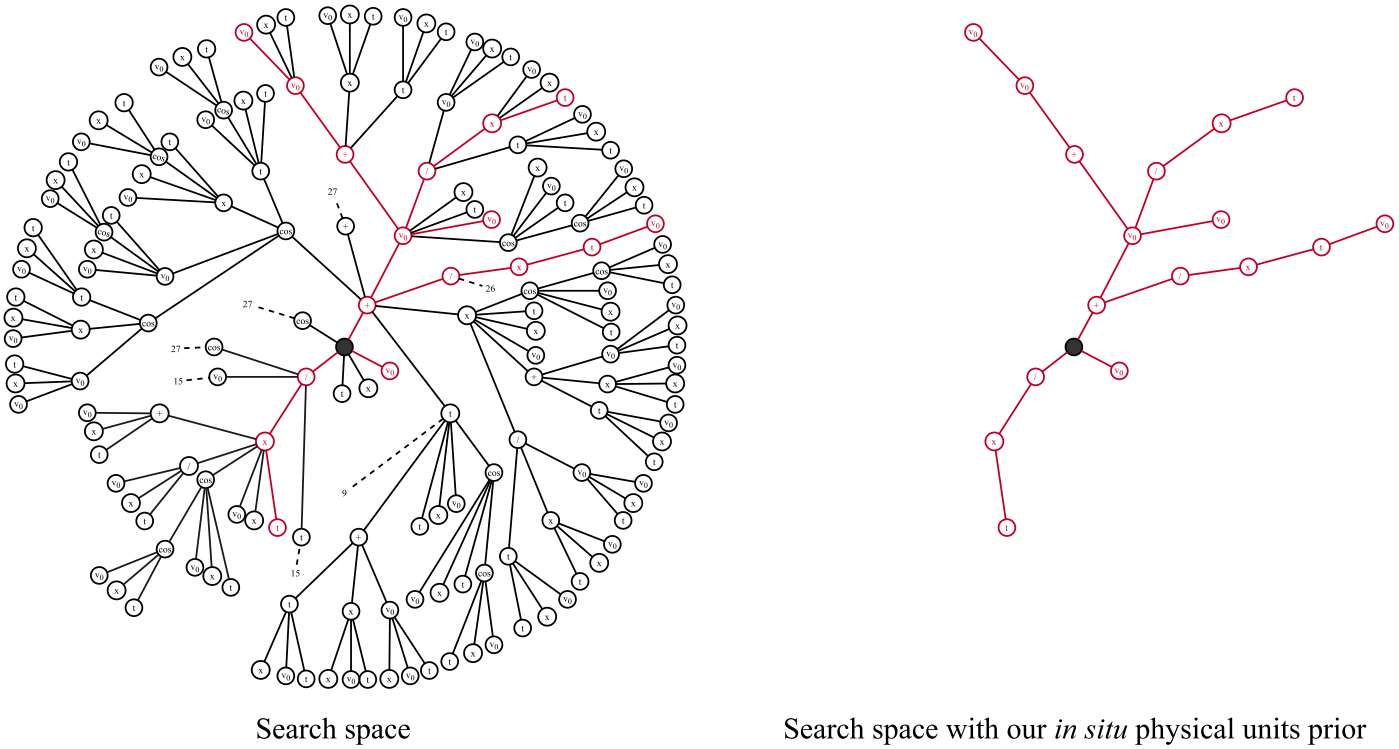

Figure 1. Illustration of the symbolic expression search space reduction enabled by our in situ physical units prior. We represent paths (in prefix notation) leading to expressions with physically possible units (in red), a sample of the paths that lead to expressions with unphysical units (in black) with other unphysical paths redacted for readability summarized with dotted lines and their total number. Here we consider the recovery of a velocity v using a library of symbols $\{+, /, \cos, v_0, x, t\}$, where v_0 is a velocity, x is a length, and t is a time (limiting ourselves to five symbol long expressions for readability). This reduces the search space from 268 expressions to only six.

incorporate and take full advantage of information on physical units during SR by storing and managing information related to dimensional analysis. This addresses, in part, the combinatorial challenge discussed above in Section 1.2. Our Φ-SO framework includes the unit constraints in situ during the equation generation process, such that only equations with balanced units are proposed by construction, thus also greatly reducing the search space as illustrated in Figure 1.

Although our framework could be applied to virtually any one of the SR approaches described in Section 2, we chose to implement our algorithm in `PyTorch` (Paszke et al. 2019, currently the most popular deep learning library in research,[6] building our method from scratch yet using some of the mathematical principles and key strategies pioneered in the state-of-the-art deep SR (DSR) framework proposed in Petersen et al. (2021a) and Landajuela et al. (2021b), which relies on reinforcement learning via a risk-seeking policy gradient (which is based on Rajeswaran et al. 2017).

In the present study, we develop a foundational symbolic embedding for physics that enables the entire expression tree graph to be tackled, as well as local units constraints. Unlike previous attempts to consider units in which data sets were rendered dimensionless before applying standard SR techniques (Udrescu & Tegmark 2020; Matchev et al. 2022; Keren et al. 2023), our approach allows us to anticipate the required units for the subsequent symbol to be generated in a partially composed mathematical expression. By adopting this approach, we not only focus on training a neural network to generate increasingly precise expressions, as in Petersen et al. (2021a), but we also generate labels of the necessary units and actively train our neural network to adhere to such constraints. In essence, our method equips the neural network with the ability to learn to select the appropriate symbol in line with local units constraints.

To the best of our knowledge, such a framework was never built before. This constitutes the first step in our planned research program of building a powerful general-purpose SR algorithm for astrophysics and other physical sciences. Our aim here is to present the algorithm to the community, show its workings, and its potential, while leaving concrete astrophysical research applications to future studies.

This study is organized as follows. We first provide a brief overview of the recent SR literature in Section 2. Our Φ-SO framework is described in detail in Section 3, in Section 4 we apply it to a benchmark of 120 equations from the Feynman Lectures on Physics (Feynman et al. 1971) and compare it to 17 other popular SR algorithms, reporting state-of-the-art performance. In Section 5, we showcase Φ-SO's capabilities on a panel of astrophysical test cases and perform an ablation study. Finally, in Sections 6 and 7, we discuss the results and draw our conclusions.

2. Related Works—A Brief Survey of Modern SR

SR has traditionally been tackled using genetic programming where a population of candidate mathematical expressions is iteratively improved through operations inspired by natural evolution such as natural selection, crossover, and mutation. This type of approach includes the well-known Eureqa software (Schmidt & Lipson 2009, 2011; see Graham et al. 2013 for a benchmark of Eureqa's capabilities on astrophysical

[6] https://paperswithcode.com/trends

test cases), as well as more recent works (Stephens 2015; Cava et al. 2019; La Cava et al. 2019; Virgolin et al. 2019; Cranmer et al. 2020; Cranmer 2020; Kommenda et al. 2020; de Franca & Aldeia 2021; Virgolin et al. 2021). In addition, SR has been implemented using various methods ranging from brute force to (un-)guided Monte Carlo, all the way to probabilistic searches (McConaghy 2011; Jin et al. 2019; Kammerer et al. 2020; Brence et al. 2021; Bartlett et al. 2023), as well as through problem simplification algorithms (Luo et al. 2022; Tohme et al. 2023).

Given the great successes of deep learning techniques in many other fields, it is not surprising that they have now been applied to SR, and now challenge the reign of Eureqa-type approaches (La Cava et al. 2021; Matsubara et al. 2022). Multiple methods for incorporating neural networks into SR have been developed, ranging from powerful problem simplification schemes (Cranmer et al. 2020; Udrescu & Tegmark 2020; Udrescu et al. 2020), to end-to-end SR methods where a neural network is trained in a supervised manner to map the relationship between data sets and their corresponding symbolic functions (Biggio et al. 2020; Aréchiga et al. 2021; Biggio et al. 2021; Alnuqaydan et al. 2022; Becker et al. 2022; d'Ascoli et al. 2022; Kamienny et al. 2022; Vastl et al. 2022; Bendinelli et al. 2023; Kamienny et al. 2023), all the way to incorporating symbols into neural networks and sparsely fitting them to enable interpretability or to recover a mathematical expression (Brunton et al. 2016; Martius & Lampert 2017; Ouyang et al. 2018; Sahoo et al. 2018; Kim et al. 2020; Panju & Ghodsi 2020; Valle & Haddadin 2021; Zheng et al. 2022). See La Cava et al. (2021), Makke & Chawla (2022), and Angelis et al. (2023) for recent reviews of SR algorithms.

While some of the aforementioned algorithms excel at generating very accurate symbolic approximations, the reinforcement learning-based deep SR framework proposed in Petersen et al. (2021a) is the new standard for exact symbolic function recovery, particularly in the presence of noise (La Cava et al. 2021; Matsubara et al. 2022). This has resulted in a number of studies in the literature built on this framework (Landajuela et al. 2021a, 2021a; Landajuela et al. 2021b; Petersen et al. 2021b; Kim et al. 2021; DiPietro & Zhu 2022; Du et al. (2022; Landajuela et al. 2022; Usama & Lee 2022; Zheng et al. 2022).

3. Method

Considering the success of deep reinforcement learning methods in accurately recovering exact symbolic expressions, which is particularly important in the field of physics where precise physical law recovery is crucial, we have chosen to incorporate this methodology into the machine-learning component of our physical SR approach. In Section 3.1, we describe how we generate analytical expressions from a recurrent neural network (RNN). Section 3.2 provides details about the algorithm we use to generate in situ units constraints, which are used to teach the RNN dimensional analysis rules and help reduce the search space. In Section 3.3, we describe the reinforcement learning strategy we adopted to make our RNN not only produce accurate expressions but also physically meaningful ones. Finally, we give computational details regarding our physical symbolic optimization implementation (`PhySO`) in Section 3.4.

3.1. Generating Symbolic Expressions

Symbolic expressions can be regarded as binary trees where each node represents a symbol of the expression in the library of available symbols, i.e., an input variable (e.g., x, t), a constant (e.g., v_0), or an operation (e.g., $+$, $-$, $\times$, $/$, sin, log, ...). In this representation, input variables and constants can be referred to as terminal nodes or symbols (having no child node), operations taking a single argument (e.g., sin, log, ...) are unary symbols (having one child node) and operations taking two arguments (e.g., $+$, $-$, $\times$, $/$, ...) are binary symbols. By considering each node first in-depth and then left to right, one can compute a one-dimensional list, i.e., a prefix[7] notation in which operators are placed before the corresponding operands in the expression, alleviating the need for parentheses. Using the prefix notation and treating symbols, referred to as tokens, as categories allows us to treat any expression as a mere sequence of categorical vectors. For example, considering a short toy library of tokens $\{+, \cos, x\}$, the operator $+$ can be encoded as [1, 0, 0], the function cos as [0, 1, 0], and the variable x as [0, 0, 1].

As in previous deep SR studies (e.g., Petersen et al. 2021a; DiPietro & Zhu 2022; Du et al. 2022; Kamienny et al. 2022; Vastl et al. 2022), treating mathematical expressions as sequences allows us to employ traditional natural language processing techniques to sample them. Token sequences are generated by using an RNN, which in essence, is a neural network that can be invoked multiple times to create a logical chain of similar operations. At each invocation $i < N$ (N representing the maximum number of steps), the RNN generates a time-dependent output and a corresponding memory state S_i. The RNN takes as input some time-dependent observations O_i[8] as well as the state of the previous call S_{i-1}. In practice, we use the RNN to generate a categorical probability distribution over the library of available tokens, which we then simply sample to draw a definite token. Once a token is generated, we feed the minimum number of tokens still needed to obtain a valid analytical expression (i.e., the number of dangling nodes), the token's properties, and the properties of its surroundings as observations for the next RNN call. Namely, we give the nature of the token that was sampled at the previous step (since the RNN does not have access to this information which is derived from a stochastic process),[9] the sibling (if any at this step) and parent tokens of the token to be generated in a tree representation, to which in the context of our Φ-SO framework we add the physical units of all of these tokens and the units required for the token to be generated so as to respect units rules. This allows the inner mechanisms of the neural network to take into account not only the local structure of the expression for generating the next token, but also to take into account local units constraints. The process described above can be repeated multiple times until a whole token function is generated in prefix notation, as illustrated in Figure 2.

It is important to note here that one can artificially tune the generated categorical distribution to incorporate prior

[7] This is also called "Polish" notation and can be converted to a tree representation or the "infix" notation, which we are more familiar with, as there is a one-to-one relationship between them.

[8] We refer to *observations* in the context of reinforcement learning, here pertaining to contextual analytical information related to the expression being generated, rather than to the scientific data being fitted.

[9] Not providing this information typically hinders performance.

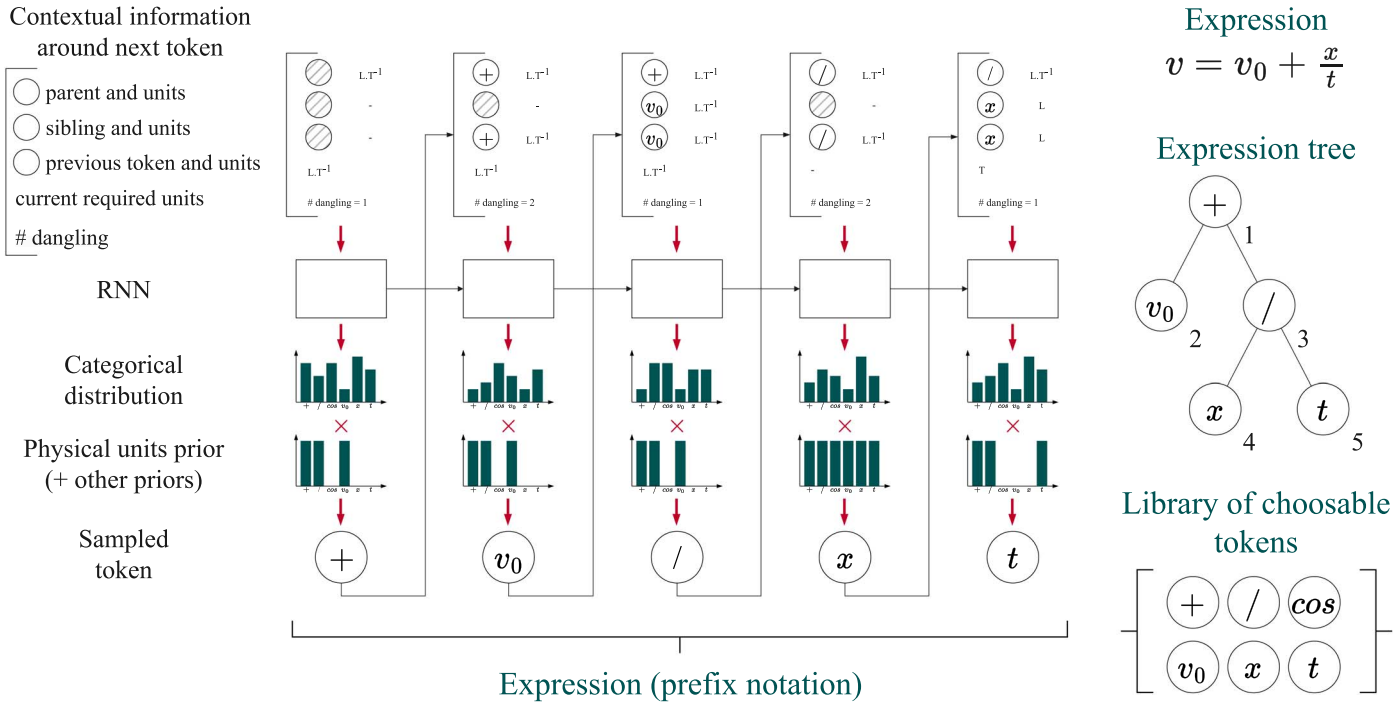

Figure 2. Expression generation sketch. The process starts at the top left RNN block. For each token, the RNN is given the contextual information regarding the surroundings of the next token to generate, namely, the parent, sibling, and previously sampled token along with their units, the required units for the token to be generated, and the dangling number (i.e., the minimum number of tokens needed to obtain a valid expression). Based on this information, the RNN produces a categorical distribution over the library of available tokens (top histograms) as well as a state that is transmitted to the RNN on its next call. The generated distribution is then masked based on local units constraints (bottom histograms), forbidding tokens that would lead to nonsensical expressions. The resulting token is sampled from this distribution, leading to the token "+" in this example. Repeating this process, from left to right, allows one to generate a complete physical expression, here [+, v_0,/, x, t], which translates into $v_0 + x/t$ in the infix notation we are more familiar with.

knowledge in situ while expressions are being generated. One can for example zero out the probability of some token depending on the context encoded in the expression tree being generated, thus greatly reducing search space (Petersen et al. 2021a, 2021b). We, therefore, adopt priors that force expression sizes to be <35 tokens long, encourage expressions to be *concise* through a soft length prior consisting of a Gaussian of variance $\sigma^2 = 5$ centered around a length of 8, to contain no more than two levels of nested trigonometric operations (e.g., forbidding $\cos(f.t + \sin(x/x0 + \tan(\square)))$ but still allowing $\cos(f.t + \sin(x/x0))$), contain no self-nesting of exponent and log operators (e.g., forbidding $e^{e^{\square}}$), and forbid useless inverse unary operations (e.g., forbidding $e^{\log\square}$). It is worth noting that the combination of priors we employ can conflict in some cases, in which case we discard the resulting candidate (e.g., the physical units prior detailed below could require a certain number of tokens to satisfy units constraints, which could conflict with the length prior requiring the expression to be terminated prematurely).

In addition to the above priors whose formulation depends on the local tree structure (parent, sibling, ancestors), our method is able to accommodate any priors that take into account the entire tree structure without having to recompute it from scratch at each step. This is rendered possible by the fact that contrary to other deep-learning-based SR algorithms, in the Φ-SO framework we compute and keep track of the full graph of the tree representation and its underlying grammatical information (such as units, symbol types like functions, free parameters, fixed constants, or the number of arguments a symbol requires), while the expression is being generated, as it is an essential ingredient to compute units constraints as detailed in the following subsection. Note that this also enables Φ-SO to accommodate any future prior relying on such information.

3.2. In Situ Physical Unit Constraints

Algorithm 1. In situ units requirement algorithm

Input: (In)-complete expression $\{\tau_j\}_{j<N}$, Position of token i
Output: Required physical units Φ_i of token at i
Function (ComputeRequiredUnits($\{\tau_j\}_{j<N}$, i))
$p \leftarrow$ PositionOfParent(i)
$s \leftarrow$ PositionOfSibling(i)
$\Phi_p \leftarrow$ Units(τ_p)
$\Phi_s \leftarrow$ Units(τ_s)
NodeRank $\leftarrow$ 1ifleftsidenodeand2ifrightnode
AdditiveTokens $\leftarrow \{+, -\}$
MultiplicativeTokens $\leftarrow \{\times, /\}$
PowerTokens $\leftarrow \{1/\square, \sqrt{\square}, \square^n\}$
PowerValues $\leftarrow \{1/\square : -1, \sqrt{\square} : 1/2, \square^n : n\}$
DimensionlessTokens $\leftarrow$ {cos,sin,tan,exp,log}
if τ_p is in AdditiveTokens and Φ_s is known **then**
$\Phi_i \leftarrow \Phi_s$
else if τ_p is in AdditiveTokens and Φ_p is free and NodeRank is 2 and Φ_s is free **then**
BottomUpUnitsAssignment(start $= s$, end $= i - 1$)
$\Phi_i \leftarrow \Phi_s$
else if Φ_p is free and τ_p is **not** in MultiplicativeTokens **and** τ_s is **not** a placeholder **then**
$\Phi_i \leftarrow$ free;
else if $i = 0$ **then**
$\Phi_i \leftarrow$ Units(root)

(Continued)

21 **else if** τ_p is in AdditiveTokens **then**
22 $\Phi_i \leftarrow \Phi_p$
23 **else if** τ_p is in PowerTokens **then**
24 $n \leftarrow$ PowerValues[τ_p]
25 $\Phi_i \leftarrow \Phi_p/n$
26 **else if** $\Phi_p = \mathbf{0}$ **or** τ_p is in DimensionlessTokens **then**
27 $\Phi_i \leftarrow \mathbf{0}$
28 **else if** τ_p is in MultiplicativeTokens **then**
29 **if** τ_i is a placeholder **and** τ_s is a placeholder **then**
30 $\Phi_i \leftarrow$ free
31 **else if** NodeRank is 1 **then**
32 $\Phi_i \leftarrow$ free
33 **else if** Φ_p is free **then**
34 $\Phi_i \leftarrow$ free
35 **else**
36 BottomUpUnitsAssignment(start $= s$, end $= i - 1$)
37 **if** τ_i is $\{\times\}$ **then**
38 $\Phi_i \leftarrow \Phi_p - \Phi_s$;
39 **else if** τ_i is $\{/\}$ **then**
40 $\Phi_i \leftarrow \Phi_s - \Phi_p$;
41 **return** Φ_i

Table 1
Dimensional Analysis Prescriptions to Enforce

Rules of Dimensional Analysis	
Expression	Units
$\tau_A \pm \tau_B$	Φ_A or Φ_B
$-\tau_A$	Φ_A
$\tau_A \times \tau_B$	$\Phi_A + \Phi_B$
τ_A/τ_B	$\Phi_A - \Phi_B$
τ_A^n	$n \times \Phi_A$
$\mathrm{op}_\mathbf{0}(\tau_A)$	$\mathbf{0}$
Rules of Units Requirements	
Expression	Requirement
$\tau_A \pm \tau_B$	$\Phi_A = \Phi_B$
$y = \tau_A$	$\Phi_y = \Phi_A$
$\mathrm{op}_\mathbf{0}(\tau_A)$	$\Phi_A = \mathbf{0}$

Note. With τ_A, τ_B, y, Φ_A, Φ_B, Φ_y referring to two nodes, the output variable and the powers of their unit vectors, $\mathrm{op}_\mathbf{0}$ denoting a dimensionless operation (e.g., {cos,sin,exp,log}), and τ_A^n representing any power operation (including, e.g., $1/\tau_A = \tau_A^{-1}$, $\sqrt{\tau_A} = \tau_A^{\frac{1}{2}}$).

Our work is part of the broader field of grammar-guided SR (Hoai et al. 2002; Manrique et al. 2009; Korns 2011; Worm & Chiu 2013; Brence et al. 2021; Ali et al. 2022; Crochepierre et al. 2022), which aims at constraining the symbolic arrangement of mathematical expressions based on domain-specific rules. Specifically and as discussed above, in physics we already know that some combinations of tokens are not possible due to units constraints. For example, if the algorithm is in the process of generating an expression in which a velocity (v_0) is summed with a length (x) divided by a token or subexpression that is still to be generated ($\square$):

$$v_0 + \frac{x}{\square}, \tag{1}$$

then based on the expression tree (as shown in Figure 2), we already know that $\square$ must be a time variable or a more complicated sub-tree that eventually ends up having units of time, but that it is definitely not a length or a dimensionless operator such as the log function.

Computing such constraints in situ, i.e., in incomplete, only partially sampled trees (containing empty placeholder nodes) is much harder than simply checking post hoc whether the units of a given equation make sense because in some situations it is impossible to compute such constraints until later on in the sequence, leaving the units of some nodes *free* (i.e., compatible with any units at this point in the sequence). For example, it is impossible to compute units requirement in the left child node of a (binary) multiplication operator token $\square \times \triangle$, as any units in the $\square$ left child node could be compensated by units in the $\triangle$ right child node. Following the dimensional analysis rules summarized in Table 1, we devised Algorithm 1. This algorithm gives the pseudo-code of the procedure we devised to compute the required units whenever possible and leaves them as free otherwise. The procedure is applied to a token at position $i < N$ in an incomplete or complete sequence of tokens $\{\tau_j\}_{j<N}$ of size N, knowing the units of terminal nodes and of the root node (e.g., respectively $\{v_0, x, t\}$ and $\{v\}$ in the example of Figure 2). The sequence may be partially made up of placeholder tokens of yet undetermined nature (representing dangling nodes). Running Algorithm 1 before each token generation step allows one to have a maximally informed expression tree graph in terms of units.

Having access in situ to the (required) physical units of tokens allows us not only to inform the neural network of our expectations in terms of units as well as to feed it units of surrounding tokens, thus allowing the model to leverage such information, but also to express a prior distribution over the library. This enables the algorithm to zero out the probability of forbidden symbols that would result in expressions that violate units rules. Combining this prior distribution with the categorical distribution given by the RNN while expressions are being generated results in a system where by construction only correct expressions with correct physical units can be formulated and learned on by the neural network.

We acknowledge a previous attempt by Udrescu & Tegmark (2020) in the `AI Feynman` algorithm to consider units in the context of SR. The approach adopted by `AI Feynman` addresses SR problems by first transforming the variables to make them dimensionless, often leading to a reduction in the number of variables and allowing the generation of physically balanced expressions. However, if this method fails, the algorithm reverts to the original problem setup. This results in `AI Feynman` resorting to fitting high-order polynomials or complicated expressions that although very accurate lack physical meaning from a dimensional analysis perspective most of the time when it is not able to find a perfect fit solution. For instance, even in the shorter range of expressions it proposes, one can find equations such as $K = \arcsin(0.169 * e^{-3.142*m+w})$, where K, m, and w denote an energy, a mass, and a velocity for Feynman problem I.13.4 (details about the Feynman SR problems can be found in Section 4.1). In contrast, Φ-SO is designed to yield only physically plausible expressions by construction all of the time. Contrary to `AI Feynman`, Φ-SO works on dimensional data by leveraging constraints on the functional forms, while generating expressions as outlined in Table 1. It is worth noting, however, that making problems dimensionless, as implemented in `AI`

`Feynman`, is a valuable approach that can work in pairs with any SR method to ensure outputs are not nonsensical.

Indeed, it could be argued that we could have tackled physical units validity of expressions in SR by taking advantage of the Buckingham Π theorem (Buckingham 1914), with variables and constants rendered dimensionless by means of multiplicative operations among them. Such an approach can actually be adopted as a preliminary step in conjunction with any SR framework (see, e.g., Matchev et al. 2022; Keren et al. 2023). However, although working with so-called Π groups ensures the generation of physically valid expressions (since all terms become dimensionless), it simultaneously removes constraints imposed by dimensional analysis, complicating the SR process. It is interesting to note that nature (or at least physics) is not dimensionless, so information is lost during the process of making variables and constants dimensionless, preventing us from leveraging the powerful constraints on the functional form associated with this dimensional information. Drawing from the example presented in Udrescu & Tegmark (2020), let us consider a data set associated with the target expression

$$F = \frac{Gm_1m_2}{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}. \tag{2}$$

When rendered dimensionless, the target expression becomes

$$y = \frac{\frac{m_2}{m_1}}{\left(\frac{x_2}{x_1} - 1\right)^2 + \left(\frac{y_2}{x_1} - \frac{y_1}{x_1}\right)^2 + \left(\frac{z_2}{x_1} - \frac{z_1}{x_1}\right)^2}. \tag{3}$$

While this transformation decreases the number of input variables to $\{\frac{m_2}{m_1}, \frac{x_2}{x_1}, \frac{y_2}{x_1}, \frac{z_1}{x_1}\}$, it simultaneously nullifies the inherent dimensional analysis constraints. Consequently, the SR algorithm could potentially produce expressions such as $\frac{m_2}{m_1} - \frac{x_2}{x_1}$ or $\left(\frac{x_2}{x_1} - 1\right)^2 + \frac{y_2}{x_1}$. In contrast, with our in situ constraints, lengths could only be summed with length terms, similarly, squared lengths could only be summed with squared lengths and having Gm_1m_2 in the numerator would be enforced by the requirement of the expression being homogeneous to a force. In essence, while rendering variables dimensionless ensures physicality of the expressions, it simultaneously relinquishes valuable constraints on their functional forms.

Finally, we note that after the first submission of our paper, two approaches similar to ours were presented, the first working in pair with a sparsity fitting method (Purcell et al. 2023) and the second working in pair with a probabilistic search method (Brence et al. 2023).

3.3. Learning

One might imagine that SR problems could be solved by directly optimizing the choice of symbols to fit the problem, using the auto-differentiation capabilities of modern machine-learning frameworks.[10] Unfortunately this approach cannot be used for SR because the cost function is nondifferentiable (the choice of selecting say the sin function over log is not differentiable with respect to the data), which prevents one from using gradient descent. A practical solution is to use a neural network as a *middleman* to generate a categorical distribution from which we can sample symbols. One can then optimize the parameters of this neural network whose task is to generate these symbols according to fit quality and physical units constraints.

Table 2
Learning Parameters

Learning Parameters	
Batch size	10,000
Learning rate	0.0025
Entropy coefficient	0.005
Risk factor	5%

The training of the network that generates the distribution of symbols relies on the *reinforcement learning* strategy (Sutton & Barto 2018), which is a common method used to train artificial intelligence agents to navigate virtual worlds such as video games,[11] or master open-ended tasks (Bauer et al. 2023). In the present context, the idea is to generate a set (usually called a *batch* in machine learning) of trial symbolic functions, and compute a scalar reward for each function by confronting it to the data. We can then require the neural network to generate a new batch of trial functions, encouraging it to produce better results by reinforcing behavior associated with high reward values, approximating gradients via a so-called *policy* (i.e., a quantitative strategy). The hope is that, by trial and error, the learnable parameters of the network will converge to values that are able to generate a symbolic function that fits the data well.

Following the insight by Petersen et al. (2021a), we adopt the risk-seeking policy gradient along with the entropy regularization scheme found by Landajuela et al. (2021b). In essence, we only reinforce the best 5% of candidate solutions, not penalizing the neural network for proposing the 95% of other candidates, therefore maximizing the reward of the few best-performing candidates rather than the average reward. With our chosen batch size of 10k, detailed in Table 2, this strategy reinforces the leading 500 candidates. This enables efficient exploration of the search space at the expense of average performance, which is of particular interest in SR as we are often mostly concerned with finding the very best candidates in particular if the goal is exact symbolic recovery and do not care if the neural network performs well on average[12]. This novel risk-seeking policy, inspired by Rajeswaran et al. (2017) and first proposed by Petersen et al. (2021a), has significantly boosted performance in SR.

It is worth noting that our approach reinforces candidates that are sampled based on not only the output of the RNN, but also local units constraints derived from the units prior distribution, which ensures the physical correctness of token choices. As a result, our approach effectively trains the RNN to make appropriate symbolic choices in accordance with local units constraints, in a quasi-supervised learning manner. This combined with the general reinforcement learning paradigm enables us to produce both accurate and physically relevant symbolic expressions.

We allow the candidate functions f to also contain *constants* with fixed physical units specified by the user, but with free

[10] Most machine-learning tasks use the differentiability of the implemented model with respect to the data to implement a (stochastic) gradient descent toward an optimal model solution that fits the data best.

[11] See, e.g., https://www.youtube.com/watch?v=QilHGSYbjDQ

[12] This is contrary to many other applications of reinforcement learning (e.g., robotic automation, video games), which can even sometimes require risk-averse gradient policies (e.g., self-driving cars; Rajeswaran et al. 2017).

numerical values. These free constants allow us the possibility to model situations where the problem has some unknown physical scales. A (somewhat contrived) example from galactic dynamics could be if we were provided a set of potential values Φ, and cylindrical coordinate values (R, z) of some mystery function that was actually a simple logarithmic potential model

$$\Phi = \frac{1}{2}v_0^2 \ln\left(R_c^2 + R^2 + \frac{z^2}{q^2}\right), \tag{4}$$

whose parameters are the velocity parameter v_0, the core radius R_c, and the potential flattening q. Of course, we will generally not know in advance either the number of such parameters that the correct solution requires, or their numerical values. Yet to be able to evaluate the loss of the trial functions f, we need to assign values to all such free *constants* they may contain. We accomplish this task by processing each trial function, with the L-BFGS (Zhu et al. 1997) optimization routine in *pytorch* (optimizing over 20 steps and using a mean squared error metric), leveraging the fact that we can encode the symbols of f using *pytorch* functions. Since *pytorch* has in-built auto-differentiation, finding the optimal value of the constants via gradient descent is extremely efficient.

Then, as in Petersen et al. (2021a) for each candidate f, we compute a reward r that is representative of fit quality: $r = 1/(1 + \text{NRMSE})$, where NRMSE is the root mean squared error normalized by the deviation of the target σ_y: $\text{NRMSE} = \frac{1}{\sigma_y}\sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - f(\boldsymbol{x}_i))^2}$. We apply the policy gradients by means of an Adam optimizer Kingma & Ba (2014) and use a long short-term memory (LSTM) type RNN (Hochreiter & Schmidhuber 1997). Our additional learning hyperparameters can be found in Table 2. It is worth noting that the empirically tuned batch size we found (10 k) is larger than the one found by Petersen et al. (2021a), which was of 1k. We attribute this to the very strong constraints offered by our Φ-SO setup, which require a strong exploration counterpart to avoid getting stuck in local minima. This helps ensure that the model does not prematurely converge by continuously reinforcing a locally optimal expression, but rather seeks more solutions until identifying the most favorable one.

It is also worth noting that in the reinforcement learning framework, the reward function can be considered as a black box, which does not have to be differentiable; therefore, one could use anything as the reward. For example, we can also include the complexity of the symbolic function in the reward function, so as to have a criterion akin to Occam's razor. But actually one could in principle implement many ideas into the reward function: symmetries, constraints on primitives or derivatives, fitness in a differential equation, the results of some symbolic computation using external packages such as `Mathematica` (Wolfram 2003) or `SymPy` (Meurer et al. 2017), behavior of the function when implemented an n-body simulation, and so on. Note that in the context of this work, although there are more sophisticated schemes to define complexity (see, e.g., Vladislavleva et al. 2009), we simply define it as length, i.e., the number of tokens appearing in the expression excluding parentheses or the number of nodes in a tree representation.

3.4. Computational Details

Due to the number of trial expressions to evaluate at each iteration and considering that each expression must be evaluated multiple times to optimize its free constants, the optimization step is one of the main performance bottlenecks of our `PhySO` algorithm. This step was therefore parallelized across the batch, resulting in a free constants optimization time of a given expression typically being of the order of the milliseconds. We show an efficiency plot in a realistic scenario in Figure 3.[13]

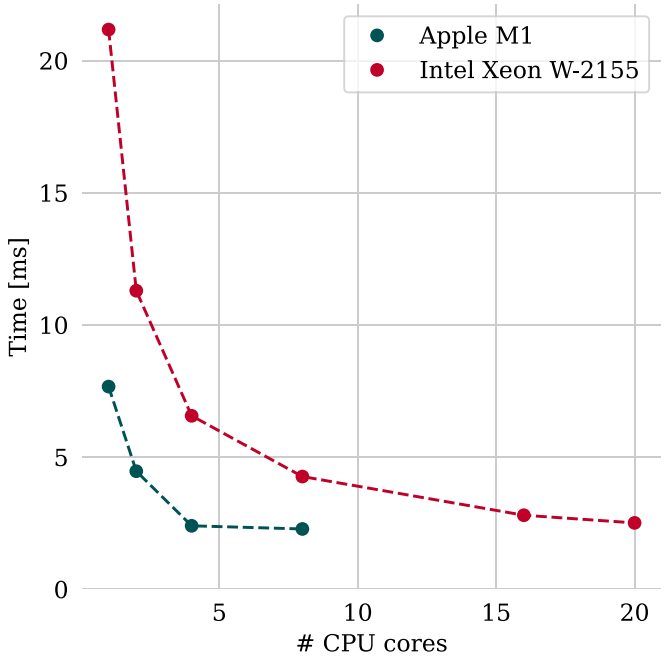

Figure 3. Computational time optimizing free constants $\{a, b\}$ in $y = a\sin(b.x) + e^{-x}$ over 20 iterations using 10^3 data points when running this task 10,000 times in parallel with our `PhySO` algorithm on an Apple M1 laptop (a machine with four fast CPU cores) and an Intel Xeon W-2155 CPU (a machine with a high number of cores).

In addition, the management of symbolic information that is necessary to compute priors and contextual information to be passed to the neural network can also occupy a non-negligible part of the computational time. In `PhySO`, these operations are therefore vectorized across both equation lengths and batches.

Lastly, it is worth mentioning that upon concluding the exploration of the equation space, `PhySO` saves Pareto front equations (optimal solutions balancing fitness and low complexity), including the overall best-fitting equation, the best-fitting equation across iterations, and stores a comprehensive log of all equations generated during the run.

4. Feynman Benchmark

To validate the efficacy of our Φ-SO method, we conducted benchmark tests using the widely recognized Feynman SR benchmark. This set of challenges, first introduced by Udrescu & Tegmark (2020) and subsequently formalized in `SRBench` (La Cava et al. 2021), encompasses 120 equations, including 100 sourced from the renowned Feynman Lectures on Physics (Feynman et al. 1971) with the other 20 sourced from other textbooks: Weinberg (1972), Goldstein et al. (2002), Jackson (2012), and Schwartz (2014). The primary objective is to

[13] The Apple M1 machine employed for the tests contains four high-performance cores and four energy-efficient cores, explaining the observed stagnation when increasing the cores count to eight.

Table 3
Summary of Baseline SR Methods along with the Underlying Techniques They Rely On

Method	Technique(s)	Description	Reference
PhySO	RL, DA	Physical symbolic optimization	This work
uDSR	RL, GP, Simp., Sup.	A unified framework for deep SR	Landajuela et al. (2022)
AIFeynman 2.0	Simp., DA	SR exploiting graph modularity	Udrescu et al. (2020)
AFP_FE	GP	AFP with co-evolved fitness estimates, Eureqa-esque	Schmidt & Lipson (2009)
DSR	RL	Deep SR	Petersen et al. (2021a)
AFP	GP	Age-fitness Pareto optimization	Schmidt & Lipson (2011)
gplearn	GP	Koza-style SR in Python	Stephens (2015)
GP-GOMEA	GP	GP-optimal mixing evolutionary algorithm	Virgolin et al. (2021)
ITEA	GP	Interaction-transformation EA	de Franca & Aldeia (2021)
EPLEX	GP	ϵ-lexicase selection	La Cava et al. (2019)
NeSymReS	Sup.	Neural SR that scales	Biggio et al. (2021)
Operon	GP	SR with nonlinear least squares	Kommenda et al. (2020)
SINDy	NeuroSym	Sparse identification of nonlinear dynamics	Brunton et al. (2016)
SBP-GP	GP	Semantic back-propagation genetic programming	Virgolin et al. (2019)
BSR	MCMC	Bayesian SR	Jin et al. (2019)
FEAT	GP	Feature engineering automation tool	Cava et al. (2019)
FFX	Rand.	Fast function extraction	McConaghy (2011)
MRGP	GP	Multiple regression genetic programming	Arnaldo et al. (2014)

Note. Reinforcement learning (RL), programming (GP), problem simplification schemes (Simp.), end-to-end supervised learning (Sup.), dimensional analysis (DA), neuro-symbolic/auto-differentiation based sparse fitting techniques (NeuroSym), Markov Chain Monte Carlo (MCMC), and random search (Rand).

retrieve these equations using only the provided data points at various levels of noise.

Although this benchmark has inherent limitations, such as treating constants of nature (e.g., G, c, $\hbar$) and discrete physical values from quantum mechanics as continuously varying input variables (which places a higher emphasis on the implementation of the problem simplification schemes developed in Udrescu & Tegmark 2020), it offers a comprehensive representation of the diversity of physical functional forms and remains a valuable standard for comparison as most SR methods have been thoroughly benchmarked on it (see La Cava et al. 2021).

Details on the benchmarking procedure can be found in Section 4.1. Results on exact symbolic recovery are provided in Section 4.2, while findings regarding fit quality are presented in Section 4.3. Finally, we provide training curves in Section 4.4.

4.1. Benchmarking Procedure

We meticulously adhered to the established protocol delineated in SRBench by La Cava et al. (2021), setting our PhySO algorithm to identify expressions that fit 10,000 data points corresponding to each Feynman benchmark equation. PhySO was only allowed to evaluate a maximum of 1 million expressions during each run and exact symbolic recovery was assessed by ensuring the difference between the expression generated by PhySO and the target expression was reduced to a constant or that the fraction simplified to a constant using the SymPy library for symbolic mathematics (Meurer et al. 2017). In addition, fit quality was assessed using the R^2 metric defined as $R^2 = 1 - \frac{\sum_{i=1}^{N}(y_i - f(\boldsymbol{x}_i))^2}{\sum_{i=1}^{N}(y_i - \bar{y})^2}$ on 100,000 noiseless test data points. As per benchmark rules, in order to ensure robustness, for each equation, this procedure was repeated multiple times (opting here for five trials over 10, due to the considerable computational demands associated with such benchmarks), each with a unique random seed, and the recovery rates were subsequently averaged. In alignment with SRBench stipulations, equations I.26.2, I.30.5, and test_10 (containing arccos and arcsin functions) as well as II.11.17 were excluded from our results. The whole benchmark tests were conducted across four noise levels: 0%, 0.1%, 1%, and 10%, leading to the evaluation of 2,320,000,000 expressions.

We ran PhySO using the hyperparameters and reward metric given in Section 3 (with the notable exception of the trigonometric prior, which was set to a maximum nesting of one) and allowing the use of $\{+,-,\times,/, 1/\square, \sqrt{\square}, \square^2, -\square, \exp,\log,\cos,\sin\}$ as well as two dimensionless adjustable free constants and a constant equal to 1 $\{\theta_1, \theta_2, 1\}$. After each run, the first few expressions (in accuracy) of the Pareto front were inspected, which proved beneficial for cases where SymPy faced simplification challenges only and made a marginal difference of approximately 1% in recovery rate. Notably, while the Feynman data set includes unit information for each variable, PhySO is the only method that capitalizes on this feature since its introduction in Udrescu & Tegmark (2020), a testament to its unique physics-specific design. For the sake of reproducibility, we provide all the code required to execute the benchmark using PhySO as well as the detailed SRBench-style results regarding each run.

We compare the performance of our Φ-SO approach to other SR algorithms with documented exact symbolic recovery rates, as reported in La Cava et al. (2021) and Landajuela et al. (2022). These algorithms are summarized in Table 3. Remarkably, this includes AFP_FE a Eureqa-like method, by the same authors combining AFP with the Eureqas method for fitness estimation (La Cava et al. 2021) and which we denote as AFP_FE ($\sim$Eureqa). In La Cava et al. (2021), DSR (Petersen et al. 2021a) was not permitted to use any free parameters when generating expressions, greatly hindering its capabilities; we therefore also consider the performance of the latest version of DSR (Landajuela et al. 2021b) self-reported in the ablation study of Landajuela et al. (2022), which relies on more suitable hyperparameters as a baseline. However, we note that is important to exercise caution when interpreting this additional DSR performance data point as well as the performances of SINDy, NeSymReS, and uDSR as our available data only offers their final scores on a composite

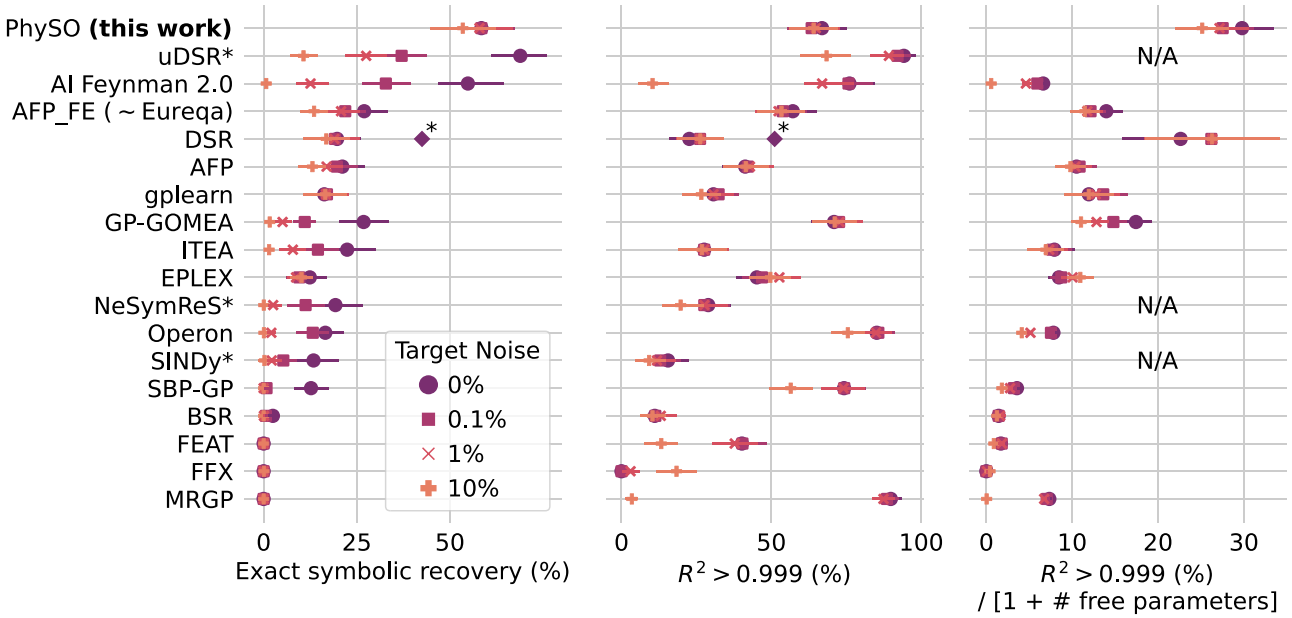

Figure 4. Exact symbolic recovery rates, rates of accurate expressions (having $R^2 > 0.999$), and rates of accurate expressions normalized by the number of free parameters appearing in expressions for PhySO and other baseline methods on the Feynman benchmark. PhySO vastly outperforms all other methods in symbolic recovery in the presence of even minimal levels of noise (>0.1%). In addition, the effectiveness of the dimensional analysis schemes of our Φ-SO approach are clearly visible when comparing DSR (a purely RL method) with our implementation: PhySO (combining RL with dimensional analysis). Error bars indicate a 95% confidence interval, ♦ denotes performances of DSR (Landajuela et al. 2021b) reported in Landajuela et al. (2022) on noiseless data with free constants allowed and * denotes that benchmarking conditions may vary and scores are polluted by approximately 5% of results from another benchmark.

data set, which encompasses both the Feynman benchmark and the Strogatz benchmark (La Cava et al. 2016)—the latter accounting for approximately 5% of the total score. This aggregated score is what we illustrate in our figures throughout this section. In addition, it is worth noting that the exact conditions under which SINDy and NeSymReS were benchmarked are unknown and that in the case of uDSR and the additional DSR data point, the benchmarking, respectively, permitted an evaluation of up to 2 million and 0.5 million expressions, respectively, in contrast to the 1 million limit set for other methods. Furthermore, detailed results for these methods, in particular those regarding the specific expressions they identified, are unavailable, preventing their inclusion in our comparative analysis when concerning expression metrics (complexity or number of free parameters). Although per SRBench rules, we permitted our method to evaluate up to 1 million expressions compared to DSR's 0.5 million, PhySO typically identifies the correct expression well before reaching this limit or not at all. Additionally, while DSR's 42% score is influenced by another benchmark, the impact is very low, accounting for only 5%. This external benchmark is relatively straightforward, with DSR achieving around 25% even without free parameters (La Cava et al. 2021), indicating its limited effect on the overall score. Thus, we believe a direct comparison between PhySO's score and DSR's from Landajuela et al. (2022) is valid, especially considering the gap in performance as detailed in the next subsection.

4.2. Exact Symbolic Recovery

Figure 4 presents the performance of PhySO against baseline algorithms from Table 3 on the Feynman benchmark. This includes the average exact symbolic recovery rate, accurate expression rate (defined as those with a fit coefficient $R^2 > 0.999$), and normalized accurate expression rate considering the number of free parameters in the expressions, across different noise levels. Compared to DSR, which strictly relies on reinforcement learning, PhySO utilizes both reinforcement learning and dimensional analysis. With DSR's score at roughly 42%, our method's 58.5% score highlights the significant benefits of incorporating dimensional analysis. In the realm of physics, the exact symbolic recovery rate is a paramount metric, and given that real-world physics data is often noisy, the resilience of an algorithm to noise is also crucial. However, with a minor noise level of 0.1%, many high-performing methods see their recovery rates almost halved. In contrast, PhySO maintains consistent performances. Remarkably, at a 10% noise level, where most methods' recovery rates dip below 20%, and even high performers like uDSR and AI Feynman 2.0 score only 10.7% and 0.7%, respectively, PhySO continues to accurately recover expressions over 53% of the time.

In noiseless scenarios, PhySO is only surpassed by uDSR, which relies on a cocktail of five of the most potent SR techniques: reinforcement learning for iterative adjustments, genetic programming for enhanced randomization and exploration, supervised learning to leverage existing knowledge, neuro-symbolic style sparse coefficient fitting for its linear symbolic modules, and powerful simplification strategies, similar to those utilized by AI Feynman 2.0, which narrowly lags behind PhySO. These techniques rely on the exploitation of separability (e.g., simplifying the search of $f(x_1, x_2)$ to the search of the simpler functions $f_1(x_1)$ and $f_2(x_2)$ with $f(x_1, x_2) = f_1(x_1) + f_2(x_2)$), symmetry (e.g., simplifying the search of $f(x_1, x_2)$ to the search of $f_1(x_1, x_2)$ and $f_2(x_2)$ with $f(x_1, x_2) = f_1(x_1, f_2(x_2))$), and many other schemes to circumvent the intricate functional forms in the benchmark. Despite relying solely on reinforcement learning and dimensional analysis, on noiseless data PhySO rivals uDSR and surpasses AI Feynman 2.0, demonstrating the effectiveness of our approach.

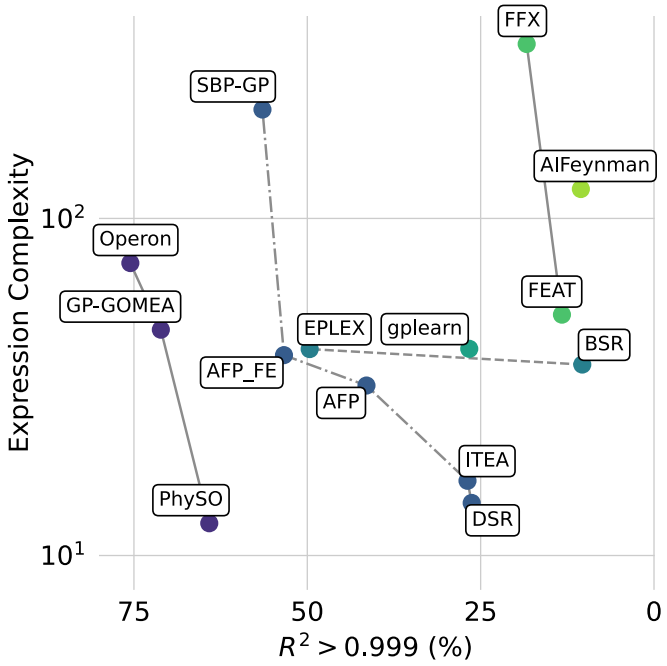

Figure 5. Complexity vs. rate of expressions having $R^2 > 0.999$ at a 10% noise level for PhySO and other SR methods from the literature on the Feynman benchmark (La Cava et al. 2021). Lines and colors denote the 1st, 2nd, 3rd, 4th, 5th, and 6th Pareto fronts following the SRBench algorithm comparison framework. PhySO is a Pareto optimum producing simple yet effective expressions.

It is worth noting that while the aforementioned AI Feynman-style *divide and conquer* simplification strategies are effective, they are extremely noise sensitive, a scenario where PhySO's approach remains stable. In summary, this benchmark shows that incorporating dimensional analysis constraints into SR significantly bolsters performance. Given the improvements shown by PhySO over DSR thanks to the inclusion of unit constraints, and given uDSR's impressive performances in noiseless scenarios, we believe combining Φ-SO with uDSR could elevate outcomes even further.

4.3. Fit Quality

Regarding the fraction of expressions with an $R^2 > 0.999$, many methods achieve high scores by incorporating an extensive number of free constants, resulting in intricate expressions that often lack interpretability and are nonsensical from a dimensional analysis standpoint. For example, AI Feynman 2.0, when not identifying the precise symbolic expressions, tends to generate complex expressions comprising, post-simplification, an average of 147 symbols, and 18 free constants due to its brute force polynomial fitting approach. Similarly, Operon[14] and MRGP expressions contain on average respectively 17 and 88 free constants post-simplification at a 10% noise level. This is not a problem in many fields where human interpretability is not a priority. However, given the importance of this criterion in physics, we also show in Figure 4 the rate of accurate expressions normalized by the number of free constants plus 1. PhySO emerges as the leading method in generating succinct, physically coherent, and interpretable expressions that best approximate a data set, that is when it is not able to recover the exact underlying expression altogether.

This is further illustrated in Figure 5, where we show Pareto frontiers of expression complexity versus fit quality at a 10% noise level for all benchmarked methods with available output expression information. On this plot, PhySO is a Pareto optimum demonstrating its ability to produce simple yet good-fitting expressions.

4.4. Learning Curves

Due to its very constraining nature, using a yet untrained neural network, our in situ units prior often conflict with the length prior which is essential to avoid the expression generation phase going on forever. This typically results in the majority of expressions being discarded due to this conflict during the first iterations of the training process. However, enabling the neural network to learn on physically correct expressions, and enabling it able to observe local units constraints, allows it to actively learn dimensional analysis rules. This is shown in Figure 6, which gives the fraction of physical expressions successfully generated over iterations of learning averaged over all runs of the Feynman benchmark at each level of noise.

Moreover, Figure 6 presents the evolution of the R^2 fit coefficient on training data for the best expression identified at each iteration. The figure demonstrates that as the iterations progress, the neural network not only improves in generating expressions with better fits but also refines its capacity to produce expressions that are physically meaningful.

In our observations, while Φ-SO occasionally escapes local minima through stochastic variations, convergence is typically characterized by the neural network mostly producing identical expressions. This state of convergence is typically reflected by both average fit quality and rate of physical expression remaining static, as well as by the reward distribution peaking. The rate of convergence is dependent on the difficulty of the case, the level of noise, and the chosen hyperparameters. As depicted in Figure 6, under the hyperparameters detailed in this study, Φ-SO typically reaches convergence well within several hundred iterations. Note that since it is operating in a reinforcement learning framework, Φ-SO is trained on *moving targets* as its targets consist of expressions generated by itself during the last iteration, which is characterized by the loss not decreasing during training except when it starts consistently producing similar equations while converging.

5. Astrophysical Case Studies

We now showcase our Φ-SO method on a panel of astrophysical test cases: the relativistic energy of a particle is examined in Section 5.1, the law describing the expansion of the Universe in Section 5.2, the isochrone action from galactic dynamics in Section 5.3, and additional toy test cases given in Section 5.4. We give the results along with an ablation study, disabling specific components of our system to determine their impact on performance, in Section 5.5. We perform this ablation study in a noiseless scenario using mock data but still demonstrate Φ-SO's abilities on observational noisy data for the case detailed in Section 5.2, showing that the method can successfully recover physical laws and relations from real or synthetic data. Mock data generation details are given in Appendix A along with units of all variables and constants

[14] It should be noted that a recent improvement of Operon (see Burlacu 2023) allowing it to produce simpler expressions was introduced after the first submission of our paper. We expect this improved version to perform better on the Feynman benchmark.

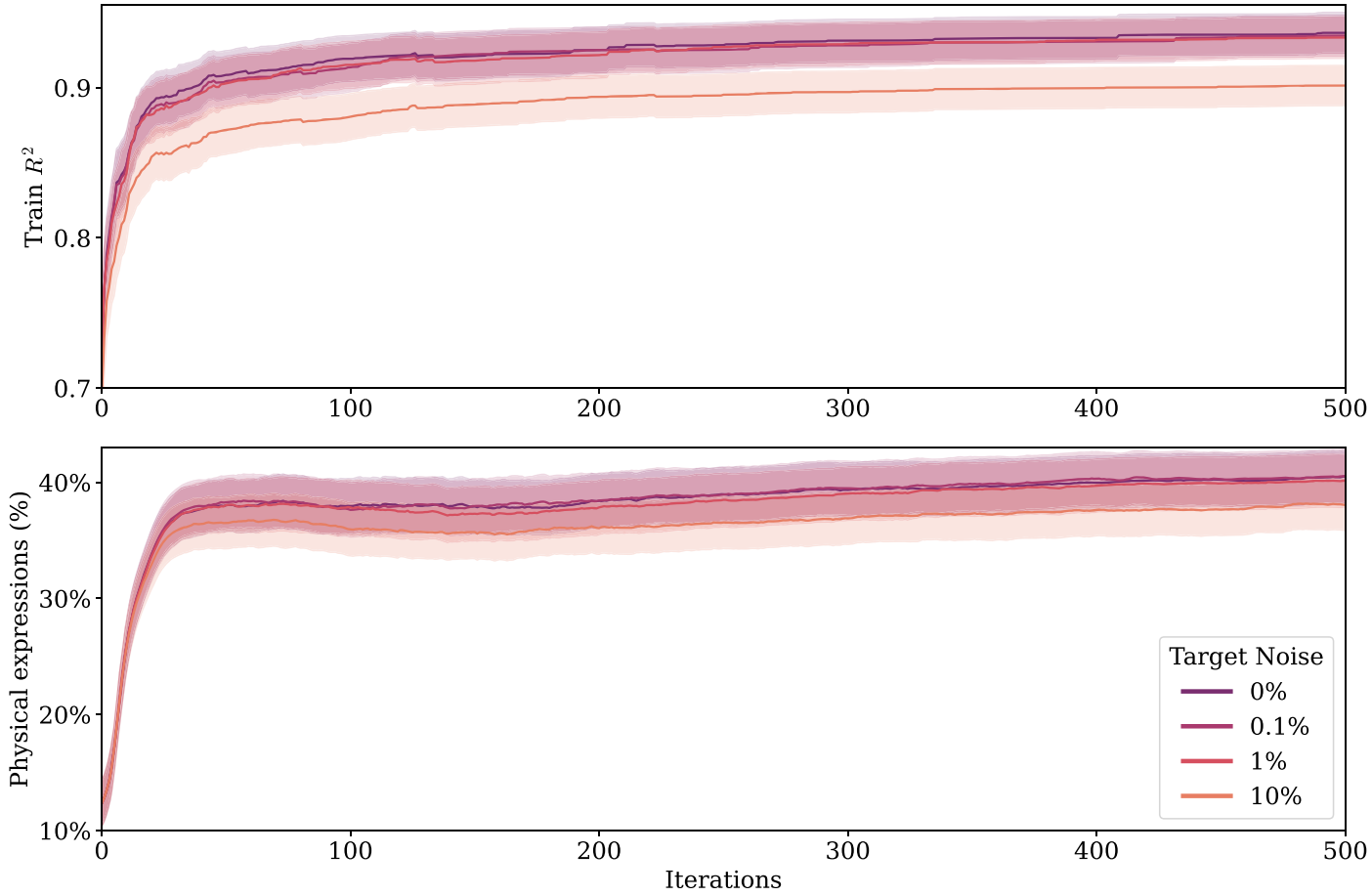

Figure 6. R^2 value on training data and percentage of expressions natively proposed by the neural network that have balanced physical units averaged across the Feynman benchmark with error regions indicating a 95% confidence interval. Φ-SO's neural network learns to produce not only good-fitting expressions but also physically meaningful ones.

involved. Note that for each of these showcases, we explicitly add the free constants described in Appendix A along with their units in Φ-SO's library of available tokens. We use the hyperparameters and reward metric detailed in Section 3 and limit ourselves to the exploration of 10 million trial expressions which roughly takes ∼4 hr (using all cores of the systems shown in Figure 3) and is only necessary for the most difficult case (the relativistic energy). In addition, for the relativistic energy showcase, we give a Pareto front, which shows the most accurate expression based on RMSE (root mean squared error) for each level of complexity. Moreover, similarly to the benchmarking in Section 4, we define the successful exact symbolic recovery of an expression by its symbolic equivalence using the `SymPy` symbolic simplification subroutine (Meurer et al. 2017). Finally, we agnostically rely on the same library of choosable tokens for all test cases: $\{+,-,\times,/,1/\square\,,\sqrt{\square}\,,\square^2\,,\exp,\log,\cos,\sin,1\}$ to which we only add input variables and free or fixed constants depending on the test cases.

5.1. Relativistic Energy of a Particle

Let us consider the expression for the relativistic energy of a particle:

$$E = \frac{mc^2}{\sqrt{1-\frac{v^2}{c^2}}}, \tag{5}$$

where m, v, and c are, respectively, the mass of the particle, its velocity, and the speed of light.

Using the aforementioned library of tokens as well as the $\{m, v\}$ input variables and a free constant $\{c\}$, Φ-SO is able to successfully recover this expression 100% of the time. Figure 7 contains the Pareto front of recovered expressions where similarly to Udrescu et al. (2020), we showcase that we are able to recover the relativistic energy of a particle as well as the classical approximation, which has a lower complexity.

However, we note that our system is able to recover the exact expression for the relativistic energy test case without any of the powerful simplification on which relies the `AI Feynman 2.0` approach proposed in (Udrescu et al. 2020; in particular, the identification of symmetries as well as the identification of additive and multiplicative separability), nor by simplifying the problem further by treating c (a constant of nature) as a variable taking a range of different values as in (Udrescu et al. 2020). Neither `DSR` (Landajuela et al. 2021b) nor `AI Feynman` (Udrescu et al. 2020) are able to crack this case under these more stringent conditions.

5.2. Expansion of the Universe

The next case study we examine is the Hubble diagram of supernovae (SNe) type Ia, namely, the change in the observed luminosity of these important standard candles as a function of redshift z. This is one of the major pieces of evidence that indicates that the Universe is experiencing an accelerating

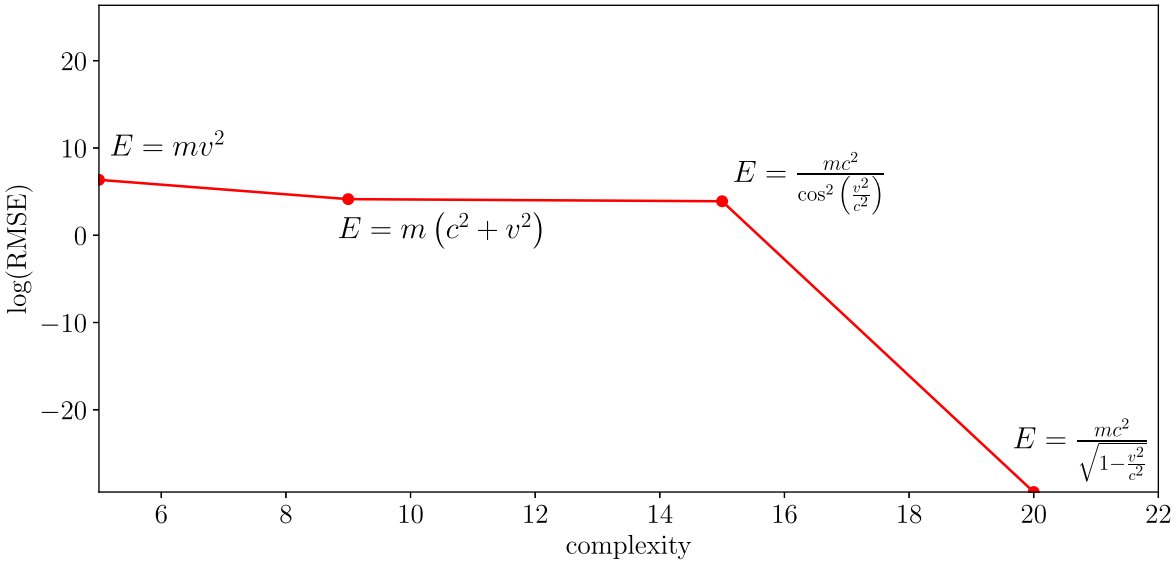

Figure 7. Pareto front encoding accuracy complexity trade-off of recovered physical formulae typically recovered using our Φ-SO method when applied to data for the relativistic energy of a particle. We recover the relativistic expression as well as the classical approximation. Note that although the exact classical expression $\frac{1}{2}mv^2$ is encountered by Φ-SO it is Pareto dominated by the simpler mv^2 expression.

expansion, and it is also one of the observational pillars underlying Lambda cold dark matter (ΛCDM) cosmology in which Dark Energy dominates the energy-density budget of the Universe.

We will use the so-called *Pantheon* state-of-the-art compilation data set (Scolnic et al. 2018), shown in Figure 8. We use a similar calibration and follow an almost identical methodology as Bartlett et al. (2023), to find the Hubble parameter $H(z)$ from the measured supernova (SN) magnitude and redshift pairs. Following Bartlett et al. (2023), we use the auxiliary function

$$y(x \equiv 1 + z) \equiv H(z)^2, \tag{6}$$

which for ΛCDM in a flat Universe with negligible radiation pressure is

$$y_{\Lambda\mathrm{CDM}}(x) = H_0^2(\Omega_m x^3 + (1 - \Omega_m)), \tag{7}$$

where Ω_m is the matter density parameter and H_0 is the Hubble constant. In a flat Universe model the cosmological luminosity distance is

$$d_L(z) = (1 + z)\int_0^z \frac{c\, dz'}{H(z')}, \tag{8}$$

where c is the speed of light.

We adapt our machinery to the Hubble diagram problem by integrating numerically the $H(z \equiv x - 1) = \sqrt{y(x)}$ functions proposed by the algorithm under Equation (8) the implied luminosity distance d_L. These are then trivially converted into a distance modulus $\mu(z) = 5\log_{10}(d_L(z)/10\ \mathrm{pc})$, which we compare to the *Pantheon* data following the procedure given in Section 3.3.

This Hubble diagram example showcases the capability of the software to include free *constants* (here we include one having the units of H_0 and the other being dimensionless as Ω_m) in the expression search, whose values are found thanks to auto-differentiation via L-BFGS optimization, as mentioned in Section 3.3. The optimal values of these constants need to be calculated after being passed through the numerical integration step (integrating Equation (8) via `PyTorch` differentiable

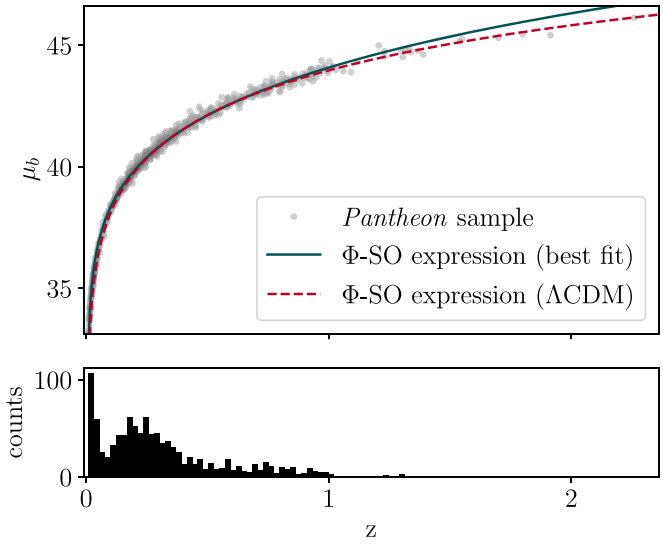

Figure 8. SR results when applying the Φ-SO algorithm (allowing two free parameters) to the Hubble diagram of SNe Ia from the *Pantheon* sample. Φ-SO rediscovers the ΛCDM relation (in red) as well as another relation (in blue), which has a slightly better fit than ΛCDM when solely considering *Pantheon*'s observational constraints due to the overabundance of low-z SNe.

cumulative trapezoids), which turns out to be the main bottleneck of the problem in terms of computational cost. However, this also shows that the algorithm allows one to derive expressions that are subsequently passed through complicated operations before being compared to data.

The Pareto front is given in Table 4 alongside the ΛCDM expression. Although we are able to recover it using synthetic data, we note that as in Bartlett et al. (2023), using observational data our system finds more accurate solutions at lower complexities than the ΛCDM model. Although this could signify that the ΛCDM theory is inaccurate, here we refrain from jumping to this conclusion because our system is only given the chance to confront its trial model of $H(z)$ to a relatively noisy data set of standard candles where there is an

overabundance of low-z events, and is not provided other observational constraints such as the cosmic microwave background, which might tilt the balance in favor of ΛCDM as the most accurate model at its level of complexity. However, although the ΛCDM expression is not the global minimum with this set of observational constraints, while exploring a space of increasingly accurate expressions our system recognizes it as an intermediate step, recording it in its history, before eventually converging to a different expression. In addition, we note that it is not surprising that our system recovers the ΛCDM expression as we allowed a maximum of two free parameters since the main goal was simply to demonstrate our system's capabilities. We defer multiparameter studies to future contributions. Finally, we are able to recover this expression by typically exploring <50,000 expressions (which takes less than a minute on the systems shown in Figure 3), the same order of magnitude as in the exhaustive SR approach proposed in Bartlett et al. (2023) but allowing more functions (cos, sin, exp, log).

5.3. Isochrone Action from Galactic Dynamics

Another interesting application of SR is to derive perfect analytical properties of analytical models of physical systems. To this end, we chose to attempt to find the radial action J_r of the spherical isochrone potential.

$$\Phi(r) = -\frac{GM}{b + \sqrt{b^2 + r^2}}, \tag{9}$$

where G is the gravitational constant, M is the mass of the model, b is the length scale of the model, and r is a spherical radius (Binney & Tremaine 2011). Action variables are special integrals of motion in integrable potentials that can be used to describe the orbit of an object in a system, and they are of particular interest in Galactic archeology as they are adiabatic invariants, so they are preserved if a galaxy or stellar system has evolved slowly. The isochrone is the only potential model to have actions known in analytic form in terms of elementary functions.[15] For the case of the isochrone model, the radial component of the action of a particle can be expressed as

$$J_r = \frac{GM}{\sqrt{-2E}} - \frac{1}{2}\left(L + \frac{1}{2}\sqrt{L^2 - 4GMb}\right), \tag{10}$$

where E and L are, respectively, the particle energy and total angular momentum (Binney & Tremaine 2011).

We provide our algorithm numerical values of J_r (which has units of angular momentum) given L and E, and leave b as a free scaling parameter. Since we expect each occurrence of M to be accompanied by an occurrence of the gravitational constant, we provide the algorithm with GM as a single variable.

This expression (Equation (10)) could not be solved either by the standard `DSR` algorithm (Landajuela et al. 2021b), or by the `AI Feynman 2.0` algorithm (Udrescu & Tegmark 2020). Our algorithm was also not able to identify the equation in 10 million guesses. However, one of the steps of the `AI Feynman 2.0` algorithm is a test for additive and multiplicative separability of the mystery function, and it creates new data sets for each separable part. For the case of additive separability, the units remain unchanged, and so it is trivial to simply provide our Φ-SO algorithm-separated data generated by `AI Feynman 2.0` to be fitted in turn, one at a time. Thus, the first term on the right-hand side of Equation (10) (with an E dependence) was easily solved together with a fitted additive-free constant. We then subtracted the fitted constant from the second data set, and Φ-SO correctly recovered the second term on the right-hand side of Equation (10) (with an L dependence).

[15] We have recently shown that actions can be calculated numerically from samples of points along orbits in realistic galaxy potentials using deep learning techniques (Ibata et al. 2021).

Table 4
Pareto Accuracy Complexity Trade-off Expressions

Expression	Complexity	$\widetilde{H}$	θ	R^2
$\widetilde{H}^2\sqrt{\theta^2 + \log(\theta + x)}$	14	5.175	−0.01	0.9955
$\widetilde{H}^2\sqrt{\theta + x}$	9	4.692	−1.01	0.9946
$\widetilde{H}^2\log(x)$	6	7.499	⋯	0.9627
$\widetilde{H}^2\log(x)^2$	8	28.276	⋯	0.9523
$\widetilde{H}^2(\theta x^3 - \theta + 1)$[a]	14	73.3	0.315	0.9166

Notes. Expressions for the auxiliary function $y(x \equiv 1 + z) \equiv H(z)^2$ applying the Φ-SO algorithm (allowing two parameters) to the Hubble diagram of SNe Ia from the *Pantheon* sample. Although Φ-SO generates the ΛCDM expression, it is not a Pareto optimum when solely considering *Pantheon*'s observational constraints due to the overabundance of low-z SNe. We include it for reference as the last line of this table.
[a] ΛCDM expression for reference.

5.4. Supplementary Cases

In addition to the cases above, we consider the following set of textbook equations for the ablation study in Section 5.5. We include Newton's law of universal gravitation:

$$F = \frac{Gm_1m_2}{r^2}, \tag{11}$$

where G is the universal gravitational constant, m_1 and m_2 are the masses of the attracting bodies, and r is the distance separating them. For this test case, we use $\{m_1, m_2, r\}$ as input variables and leave G as a free constant.

We also include a damped harmonic oscillator, which appears in a wide range of (astro)-physical contexts:

$$y = e^{-\alpha t}\cos(\omega t + \Phi), \tag{12}$$

where α and ω are, respectively, the damping parameter and the angular frequency of oscillations (both homogeneous to the inverse of a time) and Φ is the (dimensionless) phase. We leave these three parameters as free constants and use t as our input variable.

Finally, we consider a Navarro–Frenk–White (NFW) halo profile (Navarro et al. 1996), which is an empirical relation that describes the density profile $\rho(r)$ of halos of collisionless dark matter in cosmological N-body simulations:

$$\rho = \frac{\rho_0}{\frac{r}{R_s}\left(1 + \frac{r}{R_s}\right)^2}, \tag{13}$$

where r is the radius, which we use as an input variable, and ρ_0 and R_s are, respectively, the density and radius scale parameters, which we leave as free constants.

Table 5
Exact Symbolic Recovery Rate Summary and Ablation Study on Our Panel of Astrophysical Examples Using Noiseless Synthetic Data, Averaged across Five Runs

Ablation Configuration		A[a]	B	C	D[b]	E	F[c]
Physical units prior		✓	…	✓	…	✓	…
Physical units informed neural network		✓	✓	…	…	…	…
Neural network enabled		✓	✓	✓	✓	…	…
Expression	Number of expressions						
$E = \frac{mc^2}{\sqrt{1 - v^2/c^2}}$	10M	100%	0%	60%	0%	20%	0%
$J_r = \frac{GM}{\sqrt{-2E}} - \frac{1}{2}\left(L + \frac{1}{2}\sqrt{L^2 - 4GMb}\right)$	4M	100%	0%	80%	0%	60%	0%
$\rho = \rho_0 \Big/ \left(\frac{r}{R_s}\left(1 + \frac{r}{R_s}\right)^2\right)$	2M	100%	100%	40%	100%	20%	100%
$y = e^{-\alpha t}\cos(\omega t + \Phi)$	1M	100%	0%	0%	0%	0%	0%
$F = \frac{Gm_1m_2}{r^2}$	100K	100%	80%	100%	20%	80%	0%
$H^2(x \equiv 1 + z) = H_0^2(\Omega_m x^3 + (1 - \Omega_m))$	100K	100%	100%	100%	100%	40%	40%
	Average	100%	47%	63%	37%	37%	23%

Notes. By studying the performance in combinations of ablations of the in situ units prior, the neural networks's ability to be informed of local unit constraints, and of the neural network itself (i.e., replaced by a random number generator when not marked as enabled), we show that all three are essential ingredients of the success of our Φ-SO method.
[a] Full Φ-SO method.
[b] Similar to Landajuela et al. (2021b).
[c] Solely relying on a random number generator.

5.5. Ablation Study

In physics, we often seek to build approximate models, such as might be obtained via a polynomial function or a Fourier series fit to some data. In those instances, the rms error is usually the criterion of relevance to determine whether the procedure worked well or not. However, here we wish to recover the *true* underlying model, in which case the recovery rate should be the criterion of success.

The performance of Φ-SO on noiseless mock data from the test cases detailed above is summarized in the ablation study reported in Table 5. There we also report SR performance after disabling the units prior (only using the units informed RNN), disabling the RNN's ability to be informed of local units constraints (only using the units prior and a standard SR RNN), disabling both the units prior and units information (only using a standard RNN, which is similar to the Landajuela et al. 2021b setup), doing a units-guided random search by using a random number generator in lieu of the RNN, and finally doing a purely random search.

We show that merely constraining the choice of symbols using the external units prior distribution scheme (described in Section 3.2) is not enough to ensure perfect symbolic recovery of physical laws, but that informing the RNN of local units constraints (as described in Section 3.1) is essential as it allows the RNN to actively learn units rules. In addition, we show that our system does not only rely on a mere brute force approach combined with units constraints, but that the deep reinforcement learning setup described in Section 3.3 is an essential ingredient of the success of Φ-SO.

It should be noted that in the NFW test case, simply expressing the inverse of a third-degree polynomial is sufficient to solve the problem. However, using the units prior without enabling the RNN to observe local units constraints or utilizing the units prior in conjunction with a random number generator can result in a lower recovery rate compared to the use of a standalone random number generator. This is due to the highly restrictive nature of the units prior, which in a simple case like this, can actually slow down the convergence toward the solution.

Finally, we also illustrate the generalization capabilities offered by virtue of finding the exact analytical expression underlying a data set compared to a good approximation in Figure 9, where we show that such analytical expressions, as expected, vastly outperform a multilayer perceptron (MLP) neural network (here five layers of 32 units MLP having sigmoid activations and being trained until convergence on a test set, following a mean squared error loss function at 10^{-3} learning rate using an Adam optimizer, Kingma & Ba 2014).

6. Discussion

Since the deep SR framework (Petersen et al. 2021a) and most other SR methods work by maximizing fit quality, there are few constraints on the arrangement of symbols. However, the paths in fit quality and the paths in symbol arrangement toward the global minima (perfect fit quality and perfect symbol arrangement) are not necessarily correlated. This results in the curse of accuracy-guided SR, as small changes in fit quality can hide dramatic changes in functional form and vice versa. In essence, one can improve the fit quality of candidates over learning iterations while getting further away from the correct solution in symbolic arrangement. Therefore, strong constraints on the functional form, such as the one we are proposing in our setup, are of great value for guiding SR algorithms in the context of physics. This is an advantage that physics has and that Φ-SO leverages by (i) reducing the search space and (ii) enabling the neural network to actively learn dimensional analysis rules and leverage them to explore the space of solutions more efficiently. Although the possibility of making a physical units prior was hinted by Petersen et al. (2021b), to the best of our knowledge such a framework has never been built before.

The guidance offered by the units constraints gives Φ-SO an edge over other methods for finding the exact symbolic solutions, improving performance from a purely predictive

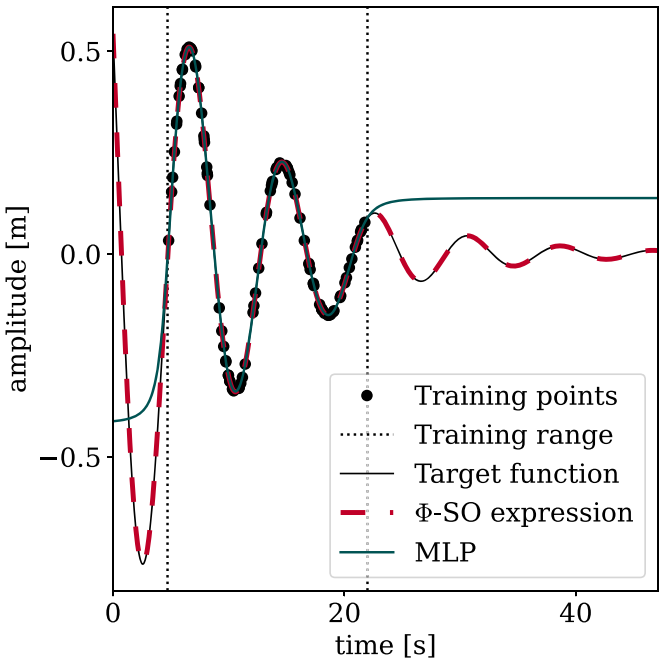

Figure 9. Example of the generalization capability of SR. Here we show randomly drawn data points (black dots) from the damped harmonic oscillator model given in Equation (12) (black line). The data are well fitted by an MLP (green line), which however fails in regions beyond the range of the training data (vertical-dotted lines). In contrast, our SR algorithm Φ-SO (red-dashed line) manages to provide much more reliable extrapolation.

standpoint. This makes Φ-SO a potentially useful tool for opening up black-box physics models such as neural networks fitted on data of physical phenomena. In addition, we note that in the context of physics, components of our Φ-SO framework can not only be used to improve the performance of algorithms built upon Petersen et al. (2021a)'s framework (Landajuela et al. 2021b; DiPietro & Zhu 2022; Du et al. 2022; Landajuela et al. 2022), but can also be used in tandem with other approaches. For instance, our in situ units prior can be used to reduce search space in the context of probabilistic or exhaustive searches (Jin et al. 2019; Kammerer et al. 2020; Brence et al. 2021; Bartlett et al. 2023), by severing physically impossible symbolic links in neuro-symbolic approaches (Brunton et al. 2016; Martius & Lampert 2017; Sahoo et al. 2018; Kim et al. 2020; Panju & Ghodsi 2020; Valle & Haddadin 2021; Zheng et al. 2022), during the seeding or mutation phases of genetic programming algorithms (Schmidt & Lipson 2009, 2011; Cava et al. 2019; La Cava et al. 2019; Virgolin et al. 2019; Cranmer et al. 2020; Cranmer 2020; de Franca & Aldeia 2021; Virgolin et al. 2021; Stephens 2015; Kommenda et al. 2020; Landajuela et al. 2022) or for making a physically motivated data set of expressions, which in conjunction with enabling the RNN to be informed of local units constraints, could improve the performance of supervised approaches (Biggio et al. 2020, 2021; Becker et al. 2022; Kamienny et al. 2022; Landajuela et al. 2022; Vastl et al. 2022; Kamienny et al. 2023).

We recognize that in its current form, Φ-SO needs to be given the physical units of the free parameters it is allowed to use. Although this is typically not an issue for SR problems that tend to fall on the more theoretical side as constants that can appear in expressions if any are usually well known, in scenarios of novel empirical scientific exploration, the appropriate selection and units of free parameters may not be immediately evident. In such scenarios, we suggest the inclusion of one free parameter for each variable, matching their units. This approach grants Φ-SO the flexibility to combine these parameters, or a subset thereof, to derive the most coherent combination that seamlessly integrates into the expression from a units perspective. As detailed in Appendix B, utilizing this protocol enables Φ-SO to accurately deduce formulae and the physical constants appearing in those. Examples include the recovery of expression for the terminal velocity during freefall, and its proportionality with the square root of an acceleration, by adeptly combining a velocity with an area to derive the acceleration parameter. In other examples, we show that Φ-SO is able to effectively rediscover the universal gravitational constant or the ideal gas constant along with their units in addition to the expressions they intervene in.

Arguably, permitting a multitude of free parameters of various physical units could inadvertently expand the search space. While this is a valid observation, it is worth noting that the algorithm remains significantly constrained, both by the limited assortment of these parameters and by the inherent units constraints between input variables, especially when considering dimensionless operations like cos,exp, and so forth. Moreover, given that the algorithm combines parameters based on the units of the variables and prioritizes solutions of lower complexity, the units of new physical constants typically align closely with the family of units of the problem, rather than assuming arbitrary values. Finally, it is worth noting that in addition to dimensional analysis constraints, another key finding of our study is that making the neural network able to observe units of symbols and currently required units in partially written expressions while they are being generated typically improves the recovery rate even without enforcing constraints directly. However, resolving SR problems without knowing a priori the units of the free parameters that can appear in the expressions is typically more difficult. We acknowledge this limitation and are actively considering future enhancements to Φ-SO that would enable it to intelligently and autonomously ascertain the units of its free parameters.

Our approach is based on a deep reinforcement learning methodology, where the neural network is reinitialized at the start of each SR task. It is therefore trained independently for each specific problem, and so does not benefit from past experience nor is it pretrained on a data set of well-known physical functional forms. One could argue that this makes our approach in principle *unbiased* akin to unsupervised learning setups and therefore well suited for discovering new physics (Karagiorgi et al. 2022). However, this also intrinsically limits SR capabilities as exploiting such prior knowledge is of great value for resolving the curse of accuracy-guided SR described above. One can exploit such prior knowledge by formulating it as an in situ prior (Guimerà et al. 2020; Kim et al. 2021) or by learning on it in a supervised manner using transformer learning techniques (Biggio et al. 2021; Kamienny et al. 2022; Vastl et al. 2022; Bendinelli et al. 2023; Kamienny et al. 2023). However, although state-of-the-art supervised SR methods, as of now, shine in providing accurate approximations, they show poorer exact symbolic expression recovery rates than other methods (see, e.g., the performances of `NeSymReS` in Figure 4 or the ablation study conducted in Landajuela et al. 2022).

While the combination of supervised and reinforcement learning may seem promising, Landajuela et al. (2022) demonstrated that such a combination offers only marginal enhancements in exact symbolic recovery. Nonetheless, in the

age of large language models, there is potential to harness vast internet-scale knowledge (see, e.g., Valipour et al. 2021). By learning the association between data points and mathematical expressions in realistic scenarios, and aligning with domain-specific assumptions using supervised learning techniques, it is conceivable to integrate this knowledge into a reinforcement learning framework, as exemplified by Fan et al. (2022). This approach might allow the recovery of expressions of substantially greater complexity than those we have explored in Section 5. In addition, while our approach generates a Pareto front that gives accuracy complexity trade-offs, future enhancements that integrate both complexity and accuracy into a singular metric (as in Bartlett et al. 2023) could potentially enhance SR performances and address model selection challenges.

As we have shown in Section 5.3, it is straightforward to improve our method by combining it with the powerful problem simplification schemes devised in Udrescu & Tegmark (2020), Udrescu et al. (2020), Luo et al. (2022), Tohme et al. (2023), and Cranmer et al. (2020). The results of the separability procedures implemented in the Udrescu et al. (2020) algorithm are conveniently recorded in separate data files, which makes it completely straightforward to use their approach as a preprocessing step for Φ-SO. We anticipate that integrating their method within our algorithm, following the approach of Landajuela et al. (2022), should enhance the performance of Φ-SO.

7. Conclusions

We have presented a new SR algorithm, built from the ground up to make use of the highly restrictive constraint that we have in the physical sciences that our equations must have balanced units. The heart of the algorithm is an embedding that generates a sequence of mathematical symbols, while cumulatively keeping track of their physical units. We adopt the very successful deep reinforcement learning strategy of Petersen et al. (2021a), which we use to train our RNN to not only produce accurate expressions but physically sound ones by making it learn local units constraints.

The algorithm was benchmarked and compared to 17 other baseline SR approaches on 120 cases from the Feynman Lectures on Physics (Feynman et al. 1971) and other textbooks. The results demonstrated the usefulness of constraints arising from dimensional analysis compared to Petersen et al. (2021a), a purely reinforcement learning-based baseline approach. In addition, our approach achieved state-of-the-art leading performances in the presence of even minimal levels of noise (exceeding 0.1%) and showing consistent performances up to 10% noise levels.

The algorithm was applied to several test cases from astrophysics. The first was a simple search for the energy of a particle in special relativity (Section 5.1), which our algorithm was able to find, yet is a problem that the standard Petersen et al. (2021a) code fails on. The second test case applied the algorithm to the famous Hubble diagram of SNe of type Ia. While the form of the Hubble parameter $H(z)$ in standard ΛCDM cosmology was indeed recovered, the algorithm finds that other simpler solutions fit the SN data (in isolation) better. This result is consistent with the findings of Bartlett et al. (2023). Another test examined a relatively complicated function in galactic dynamics, where we searched for the functional form of the radial action coordinate in an isochrone stellar potential model. This is an equation that neither the Petersen et al. (2021a) nor the Udrescu et al. (2020) methods are able to find. Although our algorithm initially failed in this test, we managed to recover the correct equation by first splitting the data set using the additive separability criterion as implemented by Udrescu & Tegmark (2020).

These tests have demonstrated the applicability of the algorithm to model data of the real world as well as to derive nonobvious analytic expressions for properties of perfect mathematical models of physical systems. Although we realize that the physical laws potentially discovered by our method will depend on data range, choice of priors, etc., this is a step toward a fully agnostic method for connecting observational data to theory. Future contributions in this research program will extend the algorithm to allow for differential and integral operators, potentially permitting the solution of ordinary and partial differential equations with physical units constraints. However, our primary goal will be to use the new machinery to discover as yet unknown physical relationships from the state-of-the-art large surveys that the astrophysical community has at its disposal.

Code Availability

The documented code for the Φ-SO algorithm along with demonstration notebooks are available on GitHub github.com/WassimTenachi/PhySO with a frozen version related to this work deposited on Zenodo (Tenachi et al. 2023).

Acknowledgments

R.I. acknowledges funding from the European Research Council (ERC) under the European Unions Horizon 2020 research and innovation program (grant agreement No. 834148). The authors would like to acknowledge the High Performance Computing Center of the University of Strasbourg for supporting this work by providing scientific support and access to computing resources. Part of the computing resources were funded by the Equipex Equip@Meso project (Programme Investissements d'Avenir) and the CPER Alsacalcul/Big Data.

Appendix A
Data Sets for the Astrophysical Examples

This appendix gives details regarding the synthetic data sets for the astrophysical examples. For each case, we generate 1000 noiseless data points following a random uniform law using arbitrary scales for the mock data. Table 6 gives the target expressions and Tables 7 and 8 give details regarding the variables and constants appearing in those expressions.

Table 6
Astrophysical Examples of Target Expressions

Case	Expression
Relativistic energy	$E = \frac{mc^2}{\sqrt{1 - v^2/c^2}}$
Isochrone action	$J_r = \frac{GM}{\sqrt{-2E}} - \frac{1}{2}\left(L + \frac{1}{2}\sqrt{L^2 - 4GMb}\right)$
NFW profile	$\rho = \rho_0 \Big/ \left(\frac{r}{R_s}\left(1 + \frac{r}{R_s}\right)^2\right)$
Damped harmonic oscillator	$y = e^{-\alpha t}\cos(\omega t + \Phi)$
Classical gravity	$F = \frac{Gm_1m_2}{r^2}$
Expansion law	$H^2(x \equiv 1 + z) = H_0^2(\Omega_m x^3 + (1 - \Omega_m))$

Table 7
Data Range and Units of the Output and Input Variables Appearing in the Astrophysical Examples

Output		Variable 1			Variable 2			Variable 3		
Name	Units	Name	Range	Units	Name	Range	Units	Name	Range	Units
E	$M.L^2.T^{-2}$	m	[−10, 10]	M	v	[−9, 9]	$L.T^{-1}$	…	…	…
J_r	$L^2.T^{-1}$	L	[2.3, 3]	$L^2.T^{-1}$	E	[−4, −6]	$M.L^2.T^{-2}$	…	…	…
ρ	$M.L^{-3}$	r	[0.2, 3]	L	…	…	…	…	…	…
y	1	t	[1.5π, 7π]	T	…	…	…	…	…	…
F	$M.L.T^{-2}$	m_1	[0, 1]	M	m_2	[0, 1]	M	r	[1, 4]	L
H^2	T^{-2}	z	[0.01, 2.5]	1	…	…	…	…	…	…

Table 8
Target Value and Units of Constants Appearing in the Astrophysical Examples

Constant 1			Constant 2			Constant 3		
Name	Value	Units	Name	Value	Units	Name	Value	Units
c	10	$L.T^{-1}$	…	…	…	…	…	…
GM	0.467	$L^3.T^{-2}$	b	1.234	L	…	…	…
r_s	1.391	L	ρ_0	0.984	$M.L^{-3}$	…	…	…
ω	0.784	T^{-1}	α	0.101	1	ϕ	0.997	1
G	1.184	$L^3.M^{-1}.T^{-2}$	…	…	…	…	…	…
H_0	1.072	T^{-1}	Ω	1.315	1	…	…	…

Appendix B
Discovering Both Analytical Laws and Constants of Nature

We note that for new scientific discovery, there are instances where the appropriate free parameters and their corresponding units are not immediately evident. In such situations, we propose a protocol wherein Φ-SO is allowed one free parameter for each input variable, sharing the same units, and another free parameter reflecting the units of the output variable. Specifically, for an SR problem consisting of the deduction of y from $\{x_1,...x_n\}$, we would permit the inclusion of $\{\theta_y, \theta_{x_1}, ... \theta_{x_n}\}$ as free constants. This grants Φ-SO the flexibility to selectively combine or omit these free parameters to construct new parameters that align with dimensional analysis constraints. In light of these combinations, we adjust the center of the soft length prior to a length of 12, facilitating longer expressions.

In this more demanding setup, we demonstrate that Φ-SO can adeptly resolve the SR challenges outlined in Table 9 (with data set details given in Tables 10 and 11), yielding both the precise symbolic expressions and their corresponding physical constants with accurate units. The scripts employed for these experiments are accessible in our repository.

For illustration, Φ-SO successfully derives the equation describing the equation of state of an ideal gas $P = C\frac{nT}{V}$ with $C = \frac{\theta_P\theta_V}{\theta_n\theta_T}$ having units $M.L^2.T^{-2}.K^{-1}.N^{-1}$ effectively rediscovering the ideal gas constant usually denoted by R. Similarly, Φ-SO is able to recover the expression for the terminal velocity of a freefalling object as a function of its mass m, its surface area A and the density of the medium it traverses ρ as $v_t = \sqrt{C\frac{m}{\rho A}}$ by unveiling its proportionality to the square root of an acceleration $\sqrt{C}$, formulated by Φ-SO as $\sqrt{\theta_v \Big/ \sqrt{\theta_A}}$, corresponding to Earth's surface gravity $\sqrt{g}$ and other scale factors. Furthermore, Φ-SO identifies the gravitational force in

Table 9
Target Expressions

Case	Expression
Ideal gas law	$P = \frac{nRT}{V}$
Freefall terminal velocity	$v_t = \sqrt{\frac{2mg}{\rho A C_d}}$
Classical gravity	$F = \frac{Gm_1m_2}{r^2}$
Blackbody photon count	$n = 1\Big/(e^{\frac{h\nu}{kbT}} - 1)$
Wave interference	$E = E_1 + E_2 + 2\sqrt{E_1E_2} + \cos\Delta\Phi$

Table 10
Data Range and Units of the Output and Input Variables Appearing in the Examples

Output		Variable 1			Variable 2			Variable 3		
Name	Units	Name	Range	Units	Name	Range	Units	Name	Range	Units
P	$L^{-1}.T^{-2}.M$	n	[1, 5]	N	T	[1, 5]	Θ	V	[1, 5]	L^3
v_t	$L.T^{-1}$	m	[1, 10]	M	ρ	[1, 6]	$M.L^{-3}$	A	[1, 5]	L^2
F	$L.T^{-2}.M$	m_1	[1, 5]	M	m_2	[1, 5]	M	r	[1, 5]	L
n	1	ν	[1, 5]	T^{-1}	T	[1, 5]	Θ	…	…	…
E	L	E_1	[1, 5]	$L^2.T^{-2}.M$	E_2	[1, 5]	$L^2.T^{-2}.M$	$\Delta\Phi$	[-5, 5]	1

Table 11
Target Value and Target Units of Constants Appearing in the Examples

Constant 1			Constant 2		
Name	Value	Units	Name	Value	Units
R	8.314	$L^{-2}.T^{-2}.M.N^{-1}.\Theta^{-1}$	…	…	…
g	9.807	$L.T^{-2}$	C_d	0.470	1
G	6.674	$L.T^{-2}.M$	…	…	…
h	6.626	$L^2.T^{-1}.M$	k_b	1.123	$L^2.T^{-2}.M.\Theta^{-1}$

relation to the involved masses m_1, m_2 and distance r as $F = Cm_1m_2/r^2$ discovering the need for a constant C having units $L^3.T^{-2}.M$ formulated by Φ-SO as $C = \theta_F\theta_r{}^2/\theta_{m_1}{}^2$, effectively rediscovering the gravitational constant G in the process. In another scenario, deriving the number density of photons recovered from a blackbody at any given temperature T and frequency ν, Φ-SO is able to recover $n = 1/(e^{\nu C/T} - 1)$, where C represents the quotient $C = h/k_b$, h, and k_b, denoting the Planck and Boltzmann constants, respectively. In most of the aforementioned cases, Φ-SO judiciously combined a subset of the available free parameters to pinpoint the precise constants needed to resolve the SR problems through a physically consistent physical law. In this last example, we show that Φ-SO recognizes scenarios where free parameters are largely redundant as it is able to derive the energy E resultant from the interference of two waves, given their energies E_1, E_2, and their phase shift $\Delta\Phi$ without the need for any of $\{\theta_E, \theta_{E_1}, \theta_{E_2}\}$.

ORCID iDs

Wassim Tenachi https://orcid.org/0000-0001-8392-3836
Rodrigo Ibata https://orcid.org/0000-0002-3292-9709
Foivos I. Diakogiannis https://orcid.org/0000-0002-8788-8174

References

Ali, M. S., Kshirsagar, M., Naredo, E., & Ryan, C. 2022, in Proc. of the Genetic and Evolutionary Computation Conf., GECCO '22 (New York: ACM), 902
Alnuqaydan, A., Gleyzer, S., & Prosper, H. 2022, MLS&T, 4, 015007
Angelis, D., Sofos, F., & Karakasidis, T. E. 2023, Arch. Comput. Methods Eng., 30, 3845
Aréchiga, N., Chen, F., Chen, Y.-Y., et al. 2021, arXiv:2112.04023
Arnaldo, I., Krawiec, K., & O'Reilly, U.-M. 2014, in Proc. of the 2014 Annual Conf. on Genetic and Evolutionary Computation, 879
Arrieta, A. B., Díaz-Rodríguez, N., Del Ser, J., et al. 2020, Inf. Fusion, 58, 82
Bartlett, D. J., Desmond, H., & Ferreira, P. G. 2023, in IEEE Transactions on Evolutionary Computation (Piscataway, NJ: IEEE), 1
Bauer, J., Baumli, K., Behbahani, F., et al. 2023, in Proc. of the 40th Int. Conf. on Machine Learning, 202, ed. A. Krause et al., 1887
Becker, S., Klein, M., Neitz, A., Parascandolo, G., & Kilbertus, N. 2022, in AI for Science: Progress and Promises (NeurIPS 2022), https://openreview.net/forum?id=vhrtZYgxLzV
Bendinelli, T., Biggio, L., & Kamienny, P. 2023, in Proc. of the 40th Int. Conf. on Machine Learning (PMLR 202), ed. A. Krause et al., 2063, https://proceedings.mlr.press/v202/bendinelli23a.html
Biggio, L., Bendinelli, T., Lucchi, A., & Parascandolo, G. 2020, in Learning Meets Combinatorial Algorithms at NeurIPS2020, https://openreview.net/pdf?id=W7jCKuyPn1
Biggio, L., Bendinelli, T., Neitz, A., Lucchi, A., & Parascandolo, G. 2021, in Proc. of the 38th Int. Conf. on Machine Learning, 139, ed. M. Meila & T. Zhang, 936, https://proceedings.mlr.press/v139/biggio21a.html
Binney, J., & Tremaine, S. 2011, Galactic Dynamics, Vol. 13 (Princeton, NJ: Princeton Univ. Press)
Brence, J., Džeroski, S., & Todorovski, L. 2023, Inf. Sci., 632, 742
Brence, J., Todorovski, L., & Džeroski, S. 2021, Knowl. Based Syst., 224, 107077
Brunton, S. L., Proctor, J. L., & Kutz, J. N. 2016, PNAS, 113, 3932
Buckingham, E. 1914, PhRv, 4, 345
Burlacu, B. 2023, in Proc. of the Companion Conf. on Genetic and Evolutionary Computation (GECCO '23) (New York: ACM), 2412
Carilli, C., & Rawlings, S. 2004, NewAR, 48, 979
Cava, W. L., Singh, T. R., Taggart, J., Suri, S., & Moore, J. 2019, in Int. Conf. on Learning Representations, https://openreview.net/forum?id=Hke-JhA9Y7
Cranmer, M., 2020 PySR: Fast & Parallelized Symbolic Regression in Python/Julia v0.2, Zenodo, doi:10.5281/zenodo.4041459
Cranmer, M., Sanchez Gonzalez, A., Battaglia, P., et al. 2020, in Advances in Neural Information Processing Systems 33 (NeurIPS 2020), 17429, https://proceedings.neurips.cc/paper_files/paper/2020/file/c9f2f917078bd2db12f23c3b413d9cba-Paper.pdf
Crochepierre, L., Boudjeloud-Assala, L., & Barbesant, V. 2022, arXiv:2202.04367
d'Ascoli, S., Kamienny, P.-A., Lample, G., & Charton, F. 2022, arXiv:2201.04600
de Franca, F. O., & Aldeia, G. S. I. 2021, Evol. Comput., 29, 367
Delgado, A. M., Wadekar, D., Hadzhiyska, B., et al. 2022, MNRAS, 515, 2733
Desmond, H., Bartlett, D. J., & Ferreira, P. G. 2023, MNRAS, 521, 1817
DiPietro, D. M., & Zhu, B. 2022, arXiv:2209.01521
Du, M., Chen, Y., & Zhang, D. 2022, arXiv:2210.02181
Fan, L., Wang, G., Jiang, Y., et al. 2022, in Thirty-sixth Conf. on Neural Information Processing Systems Datasets and Benchmarks Track, https://openreview.net/forum?id=rc8o_j8I8PX
Feynman, R. P., Leighton, R. B., Sands, M., et al. 1971, The Feynman Lectures on Physics (Reading, MA: Addison-Wesley) 1
Gaia Collaboration, Prusti, T., De Bruijne, J., et al. 2016, A&A, 595, A1
Galilei, G. 1623, Il Saggiatore (Rome: Giacomo Mascardi)
Goldstein, H., Poole, C., & Safko, J. 2002, Classical Mechanics (San Francisco, CA: Addison-Wesley)
Graham, M. J., Djorgovski, S., Mahabal, A. A., Donalek, C., & Drake, A. J. 2013, MNRAS, 431, 2371
Guimerà, R., Reichardt, I., Aguilar-Mogas, A., et al. 2020, SciA, 6, eaav6971
Hoai, N., McKay, R., Essam, D., & Chau, R. 2002, in Proc. of the 2002 Congress on Evolutionary Computation (CEC'02), 1326
Hochreiter, S., & Schmidhuber, J. 1997, Neural Comput., 9, 1735
Ibata, R., Diakogiannis, F. I., Famaey, B., & Monari, G. 2021, ApJ, 915, 5
Jackson, J. 2012, Classical Electrodynamics (New York: Wiley)
Jin, Y., Fu, W., Kang, J., Guo, J., & Guo, J. 2019, arXiv:1910.08892
Kamienny, P., Lample, G., Lamprier, S., & Virgolin, M. 2023, in Proc. of Machine Learning Research, 202, Int. Conf. on Machine Learning, ICML 2023, ed. A. Krause et al., 15655, https://proceedings.mlr.press/v202/kamienny23a.html
Kamienny, P.-A., d'Ascoli, S., Lample, G., & Charton, F. 2022, in Advances in Neural Information Processing Systems, ed. A. H. Oh et al., https://openreview.net/forum?id=GoOuIrDHG_Y

Kamienny, P.-A., & Lamprier, S. 2022, in AI for Science: Progress and Promises (NeurIPS 2022), https://openreview.net/forum?id=yeF6cyYU7W
Kammerer, L., Kronberger, G., Burlacu, B., et al. 2020, Genetic Programming Theory and Practice XVII (Berlin: Springer), 79
Karagiorgi, G., Kasieczka, G., Kravitz, S., Nachman, B., & Shih, D. 2022, NatRP, 4, 399
Keren, L. S., Liberzon, A., & Lazebnik, T. 2023, NatSR, 13, 1249
Kim, J. T., Landajuela, M., & Petersen, B. K. 2021, in 1st Mathematical Reasoning in General Artificial Intelligence, Int. Conf. on Learning Representations (ICLR), https://www.osti.gov/servlets/purl/1782518
Kim, S., Lu, P. Y., Mukherjee, S., et al. 2020, IEEE Trans. Neural Netw. Learn. Syst., 32, 4166
Kingma, D., & Ba, J. 2014, arXiv:1412.6980
Kommenda, M., Burlacu, B., Kronberger, G., & Affenzeller, M. 2020, Genet. Program. Evolvable Mach., 21, 471
Korns, M. F. 2011, in Genetic Programming Theory and Practice VIII (New York: Springer), 109
La Cava, W., Danai, K., & Spector, L. 2016, Eng. Appl. Artif. Intell., 55, 292
La Cava, W., Helmuth, T., Spector, L., & Moore, J. H. 2019, Evol. Comput., 27, 377
La Cava, W., Orzechowski, P., Burlacu, B., et al. 2021, in Proc. of the Neural Information Processing Systems Track on Datasets and Benchmarks, ed. J. Vanschoren & S. Yeung, https://datasets-benchmarks-proceedings.neurips.cc/paper_files/paper/2021/file/c0c7c76d30bd3dcaefc96f40275bdc0a-Paper-round1.pdf
Landajuela, M., Lee, C. S., Yang, J., et al. 2022, in Advances in Neural Information Processing Systems 35 (NeurIPS 2022), 33985, https://proceedings.neurips.cc/paper_files/paper/2022/file/dbca58f35bddc6e4003b2dd80e42f838-Paper-Conference.pdf
Landajuela, M., Petersen, B. K., Kim, S., et al. 2021a, in Proc. of the 38th Int. Conf. on Machine Learning, 139, ed. M. Meila & T. Zhang, 5979, https://proceedings.mlr.press/v139/landajuela21a.html
Landajuela, M., Petersen, B. K., Kim, S. K., et al. 2021b, in 1st Mathematical Reasoning in General Artificial Intelligence, Int. Conf. on Learning Representations (ICLR), https://www.osti.gov/servlets/purl/1890084
Laureijs, R., Amiaux, J., Arduini, S., et al. 2011, arXiv:1110.3193
Lemos, P., Jeffrey, N., Cranmer, M., Ho, S., & Battaglia, P. 2022, MLS&T, 4, 045002
Liu, Z., & Tegmark, M. 2021, PhRvL, 126, 180604
Liu, Z., Wang, B., Meng, Q., et al. 2021, PhRvE, 104, 055302
LSST Science Collaboration, Abell, P. A., Allison, J., et al. 2009, arXiv:0912.0201
Lu, Z., Pu, H., Wang, F., Hu, Z., & Wang, L. 2017, in Advances in Neural Information Processing Systems 30 (NIPS 2017), https://proceedings.neurips.cc/paper_files/paper/2017/file/32cbf687880eb1674a07bf717761dd3a-Paper.pdf
Luo, C., Chen, C., & Jiang, Z. 2022, Int. J. Comput. Methods, 19, 2142002
Makke, N., & Chawla, S. 2022, arXiv:2211.10873
Manrique, D., Ríos, J., & Rodríguez-Patón, A. 2009, Encyclopedia of Artificial Intelligence (Hershey, PA: IGI Global), 767
Martius, G., & Lampert, C. H. 2017, Extrapolation and Learning Equations, https://openreview.net/forum?id=BkgRp0FYe
Matchev, K. T., Matcheva, K., & Roman, A. 2022, ApJ, 930, 33
Matsubara, Y., Chiba, N., Igarashi, R., & Ushiku, Y. 2022, in AI for Science: Progress and Promises, NeurIPS 2022, https://openreview.net/forum?id=oKwyEqClqkb
McConaghy, T. 2011, Genetic Programming Theory and Practice IX (Berlin: Springer), 235
Meurer, A., Smith, C. P., Paprocki, M., et al. 2017, PeerJ Comput. Sci., 3, e103
Murdoch, W. J., Singh, C., Kumbier, K., Abbasi-Asl, R., & Yu, B. 2019, PNAS, 116, 22071
Navarro, J. F., Frenk, C. S., & White, S. D. M. 1996, ApJ, 462, 563
Ouyang, R., Curtarolo, S., Ahmetcik, E., Scheffler, M., & Ghiringhelli, L. M. 2018, PhRvM, 2, 083802
Panju, M., & Ghodsi, A. 2020, arXiv:2011.02415
Paszke, A., Gross, S., Massa, F., et al. 2019, in Advances in Neural Information Processing Systems 32 (NeurIPS 2019), https://proceedings.neurips.cc/paper_files/paper/2019/file/bdbca288fee7f92f2bfa9f7012727740-Paper.pdf
Petersen, B. K., Larma, M. L., Mundhenk, T. N., et al. 2021a, in Int. Conf. on Learning Representations, https://openreview.net/forum?id=m5Qsh0kBQG
Petersen, B. K., Santiago, C., & Landajuela, M. 2021b, in 8th ICML Workshop on Automated Machine Learning (AutoML), https://openreview.net/forum?id=yAis5yB9MQ
Press, W. H., Teukolsky, S. A., Vetterling, W. T., & Flannery, B. P. 2007, Numerical Recipes: The Art of Scientific Computing (3rd ed.; Cambridge: Cambridge Univ. Press)
Purcell, T. A., Scheffler, M., & Ghiringhelli, L. M. 2023, JChPh, 159, 114110
Rajeswaran, A., Ghotra, S., Ravindran, B., & Levine, S. 2017, in Int. Conf. on Learning Representations, https://openreview.net/forum?id=SyWvgP5el
Reinbold, P. A., Kageorge, L. M., Schatz, M. F., & Grigoriev, R. O. 2021, NatCo, 12, 3219
Sabbatini, F., & Calegari, R. 2022, arXiv:2211.00238
Sahoo, S., Lampert, C., & Martius, G. 2018, in Proc. of the 35th Int. Conf. on Machine Learning, 80, 4442, https://proceedings.mlr.press/v80/sahoo18a.html
Schmidt, M., & Lipson, H. 2009, Sci, 324, 81
Schmidt, M., & Lipson, H. 2011, Age-Fitness Pareto Optimization (New York: Springer), 129
Schwartz, M. 2014, Quantum Field Theory and the Standard Model (Cambridge: Cambridge Univ. Press)
Scolnic, D. M., Jones, D. O., Rest, A., et al. 2018, ApJ, 859, 101
Shao, H., Villaescusa-Navarro, F., Genel, S., et al. 2022, ApJ, 927, 85
Stephens, T. 2015, GPLearn, https://gplearn.readthedocs.io/en/stable/index.html
Sutton, R. S., & Barto, A. G. 2018, Reinforcement Learning: An Introduction (Cambridge, MA: MIT Press)
Tenachi, W., Ibata, R., & Diakogiannis, F., 2023 PhySO-v1.0.0, Zenodo, doi:10.5281/zenodo.8415435
Tohme, T., Liu, D., & Youcef-toumi, K. 2023, in Transactions on Machine Learning Research, 539, https://openreview.net/forum?id=lheUXtDNvP
Udrescu, S.-M., Tan, A., Feng, J., et al. 2020, in Advances in Neural Information Processing Systems 33 (NeurIPS 2020), 4860, https://proceedings.neurips.cc/paper_files/paper/2020/file/33a854e247155d590883b93bca53848a-Paper.pdf
Udrescu, S.-M., & Tegmark, M. 2020, SciA, 6, eaay2631
Usama, M., & Lee, I.-Y. 2022, Senso, 22, 8240
Valipour, M., You, B., Panju, M., & Ghodsi, A. 2021, arXiv:2106.14131
Valle, C. M. C., & Haddadin, S. 2021, arXiv:2105.14396
Vastl, M., Kulhánek, J., Kubalík, J., Derner, E., & Babuška, R. 2022, arXiv:2205.15764
Virgolin, M., Alderliesten, T., & Bosman, P. A. 2019, in Proc. of the Genetic and Evolutionary Computation Conf., 1084
Virgolin, M., Alderliesten, T., Witteveen, C., & Bosman, P. A. N. 2021, Evol. Comput., 29, 211
Virgolin, M., & Pissis, S. P. 2022, Transactions on Machine Learning Research, https://openreview.net/forum?id=LTiaPxqe2e
Vladislavleva, E. J., Smits, G. F., & den Hertog, D. 2009, IEEE Trans. Evol. Comput., 13, 333
Wadekar, D., Thiele, L., Villaescusa-Navarro, F., et al. 2023, PNAS, 120, e2202074120
Wadekar, D., Villaescusa-Navarro, F., Ho, S., & Perreault-Levasseur, L. 2020, arXiv:2012.00111
Weinberg, S. 1972, Gravitation and Cosmology: Principles and Applications of the General Theory of Relativity (New York: Wiley)
Wilstrup, C., & Kasak, J. 2021, arXiv:2103.15147
Wolfram, S. 2003, The Mathematica Book, Vol. 1 (Cambridge: Cambridge Univ. Press)
Wong, K. W., & Cranmer, M. 2022, in Proc. of the Thirty-ninth Int. Conf. on Machine Learning (ICML 2022), Machine Learning for Astrophysics, 25
Worm, T., & Chiu, K. 2013, Proc. of the 15th Annual Conf. on Genetic and Evolutionary Computation, GECCO '13 (New York: ACM), 1021
Wu, T., & Tegmark, M. 2019, PhRvE, 100, 033311
Željko, I., Kahn, S. M., Tyson, J. A., et al. 2019, ApJ, 873, 111
Zheng, W., Sharan, S., Fan, Z., et al. 2022, arXiv:2212.14849
Zhu, C., Byrd, R. H., Lu, P., & Nocedal, J. 1997, ACM Trans. Math. Softw., 23, 550